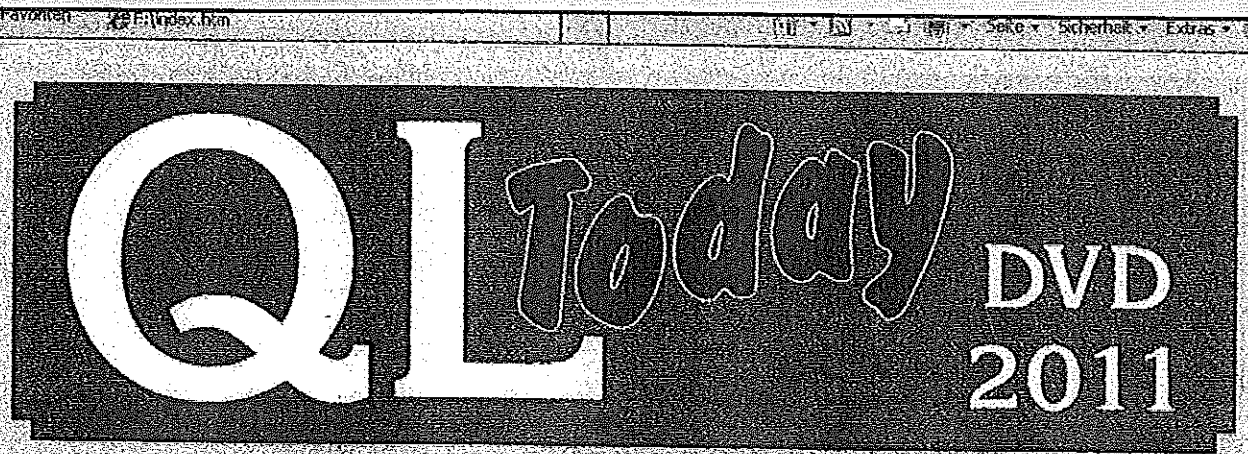
QL Today

Volume 16
Issue 1
Sept. - Nov.
2011

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...



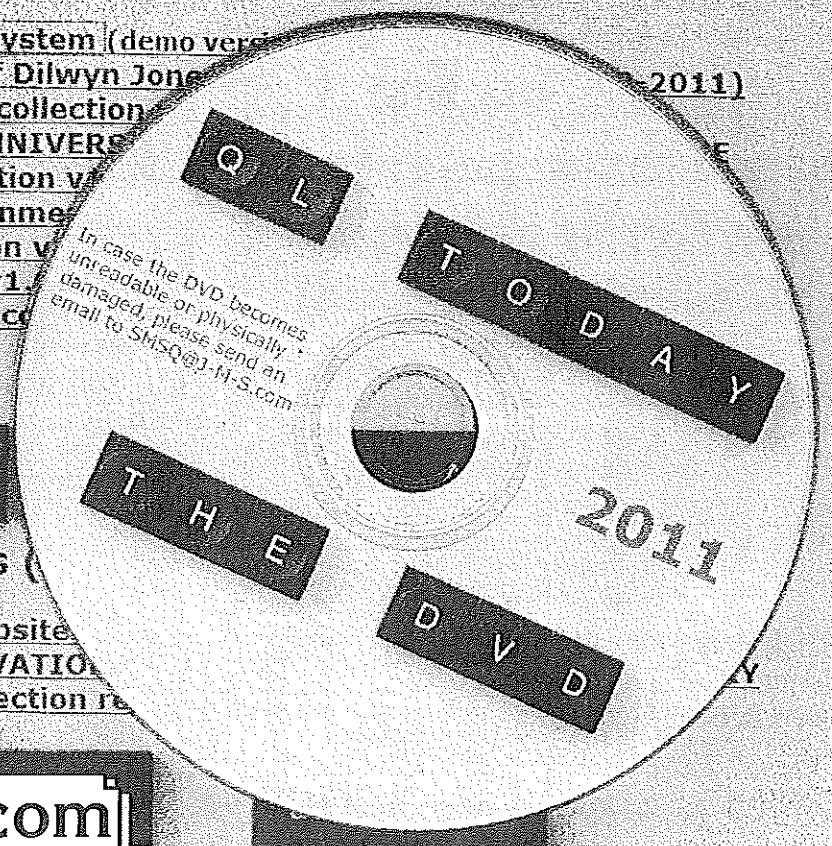
QL Today Volume 1 to 15 English (scanned PDF by Rainer Wolkwitz)

QL Today Volume 1 to 5 Deutsch (scanned PDF by Rainer Wolkwitz)

T H E D V D

Bonus Material

- QPC2, the virtual QL system (demo ver
- Local (offline) copy of Dilwyn Jone -2011)
- QL DOCUMENTATION collection
- QL HISTORY, 25TH ANNIVERS
- QL EMULATORS collection v
- QL ON A STICK environme
- QL TOOLKITS collection v
- QL GAMES collection v1.
- QL PICTURE GALLERY c



Q L

Web Links

- Dilwyn Jones's QL website
- SINCLAIR QL PRESERVATIO
  (live link, online connection r

www.QLToday.com

Contents

3 Editorial

4 News

7 64-Bit Arithmetic *George Gwilt*

13 Lessons from Statistics *Geoff Wicks*

14 Getting the table of contents of an Audio CD from the Internet under QPC - Part 1 *Wolfgang Lenerz*

19 More Assembler Discussions *George Gwilt & Norman Dunbar*

22 Glossary of Abbreviations and Terms Part 1 - 1 to 9, A to C *Dilwyn Jones & Lee Privett*

25 Small Ads

26 I2C Interface for QL Emulators - Part 2 *Ian Burkinshaw*

34 The QL Today DVD 2011

34 Vienna 2012

The deadline for the next issue is the 15th of November 2011

In case the DVD becomes unreadable or physically damaged, please send an email to SMSQ@J-M-S.com

Advertisers

in alphabetical order

Jochen Merz Software 34-35
Quanta 17
QuoVadis Design 21

QL Today

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software
Kaiser-Wilhelm-Str. 302
47169 Duisburg
Germany

Tel. +49 203 502011
Fax +49 203 502012
email: smsq@j-m-s.com
email: QLToday@j-m-s.com

Editor:

Geoff Wicks
Flat 5b
Wordsworth Avenue
Derby DE24 9HQ
United Kingdom

Tel. +44 1332 271366
email: gtwicks@btinternet.com
email: QLToday@j-m-s.com

Co-Editor & UK Office:

Bruce Nicholls
38 Derham Gardens
Upminster
Essex RM14 3HA
United Kingdom

Tel +44 20 71930539
Fax +44 870 0568755
email: qltoday@q-v-d.demon.co.uk
email: QLToday@j-m-s.com

QL *Today* is published four times a year, our volume begins on beginning of June. Please contact the German or English office for current subscription rates or visit our homepage www.QLTODAY.com.

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

QL *Today* reserves the right to publish or not publish any material submitted. Under no circumstances will **QL *Today*** be held liable for any direct, indirect or consequential damage or loss arising out of the use and/or inability to use any of the material published in **QL *Today***. The opinions expressed herein are those of the authors and are not necessarily those of the publisher.

If you need more information about the UNZIP program which is used by our BOOT program to unpack the files, we suggest that you visit Dilwyn Jones' web site where you find more information about lots of interesting QDOS software and INFOZIP at http://www.dilwyn.uk6.net/arch/index.html

Editorial

by Geoff Wicks

This is a historic issue of QL Today because it comes with a DVD containing the complete 15 year archive.

Who would have thought when QL Today was first published in 1996 that it would still be in existence 15 years later? Many thought the QL would not survive that long.

In 1996 the omens for QL Today were not good. QL World had ceased publication less than two and a half years previously and two alternative publications, QReview and IQLR, had also closed down. Against seemingly unfavourable odds QL Today survived and in 2005 became, with the notable exception of the Quanta Magazine, the longest continuously published QL magazine.

In the 15 years of QL Today's life the QL has made great strides. The first issue of QL Today contained a report of a Quanta Workshop in Tynemouth. The big news was the imminent release of a graphics card, to be called the "Aurora", to liberate QL-ers from the constraints of a 512 x 256 screen. Elsewhere in the magazine was an article by Albin Hessler about the imminent release of a software program that many had said was impossible. It would be a QL emulator for a PC and would be called "QPC".

There is an essential accessory to go with your archive DVD. Brian Kemmett has compiled a complete index of 15 years of QL Today that covers 73 pages, equivalent to more than two issues of the magazine. Think then of how many words have been written about the QL; how many topics have been covered; and how many people have contributed. All this has been voluntary. Not one writer has been paid for his work.

QL Today relies for its existence on the loyalty of its readers and not just those who write for us. Our archive DVD became possible because of readers who helped us out. Norman Dunbar researched the possibilities of an electronic QL Today, Urs König scanned a sample issue to test the feasibility and Rainer Wolkwitz scanned all 15 years. Throughout QL Today's history Brian Kemmett has faithfully prepared indexes that I have found invaluable. I frequently have to research past issues and Brian has saved me hours of work.

Earlier in this editorial I mentioned that the Quanta magazine has survived longer than QL Today. Quanta has the distinction, narrowly rivalled by JMS, of being the longest actively surviving part of the QL community. Although it has survived as long as the QL itself, it is now going through a rough period in which its future is endangered. In the worst case scenario Quanta will have started winding up proceedings by this time next year.

Quanta is making strenuous efforts to safeguard its future, but some of the changes cannot take effect immediately. The next elections must take place under the present constitution and the chairman cannot stand for re-election having served 7 years on the committee. If no replacement can be found Quanta has three months to solve the problem before commencing winding up proceedings.

Next year will be a tough year for Quanta with uncertainty over the chairman; uncertainty over the effect of a 42% rise in the subscription; and uncertainty over a revised constitution. The committee may well have to resort to emergency procedures to protect Quanta during this difficult period.

More than anything else Quanta now needs its members to show it the same active loyalty as our readers have done to QL Today.

QUANTA'S Six Year Dilemma

Quanta members must decide in the next four months whether to scrap a controversial ruling that committee members have to step down after serving continuously for six years on the committee. Earlier this year several members called for the rule to be abolished, but the committee are to recommend its continuance.

Last year a crisis over a lack of nomination for the post of treasurer resulted in the committee co-opting the previous treasurer for a further year even though he had already served 7 years on the committee. In February a discussion, sometimes heated, took place on the QL Users email group when questions were raised about the legality of this co-option. The committee conceded the relevant sections of the constitution were unduly complicated and ambiguous and embarked on a major revision of the constitution.

A further crisis could occur early next year when chairman, *Sarah Gilpin*, has to step down having served 7 years on the committee. If no successor can be found then Quanta has to commence proceedings to wind itself up in July of next year.

The six year rule was approved in 2005 and arose from the time when Quanta, according to two of its former officers, had reached its lowest ebb. The then chairman was publicly expressing opinions that there was little future for Quanta and the QL, but continued to stand for office in the absence of other candidates. The organisation was allowed to drift and was losing membership at the rate of almost 10% per year.

Since the implementation of the rule Quanta's fortunes have improved. Membership has stabilised and the committee has increased in strength. In particular the Quanta Magazine has improved considerably, albeit under a long temporary editorship. There has been a much improved news coverage and a helpline has become a permanent feature. Quanta has also successfully launched an electronic edition. However there have also been disappointments. Quanta is now able to run only one show a year, with attendance barely touching 20, and for a over a year has had a non-functioning website.

The people who have worked to restore the fortunes of Quanta are now reaching the end of their permitted terms of office. *John Gilpin* had to step down last year and *Sarah Gilpin* next year. *Dan Abbott* and *Dilwyn Jones* follow in 2014 and *Alison Southern* a year later. Those members who are opposed to the six year rule argue committee members should be able to serve for as long as they wish to avoid regular crises. The committee argue that the six year rule is there for a good reason and that if Quanta is to further develop then fresh blood is necessary. They also point out that abolishing the six year rule does not solve the problem of the unwillingness of members to take a part in the running of Quanta.

Quanta members are being asked to decide before the end of this year whether the six year rule should remain as part of a consultation on a draft constitution which members will receive with the Quanta Magazine. The draft has been drawn up by *Rich Mellor*, a former solicitor and company secretary, with the assistance of *Geoff Wicks*, who has law court and works council experience. In late July both attended a meeting of the Quanta Committee to act as advisors in a discussion on the constitution.

Geoff Wicks comments:

"We did not regard it as being in our remit to make decisions on behalf of the members. Our job was to go through the constitution clause by clause looking for weak points and obscure or ambiguous wording. We have tried to bring the draft constitution better into line with Quanta's present way of working. There are still some important decisions to be made by the members before a final draft is prepared early next year for consideration at the AGM"

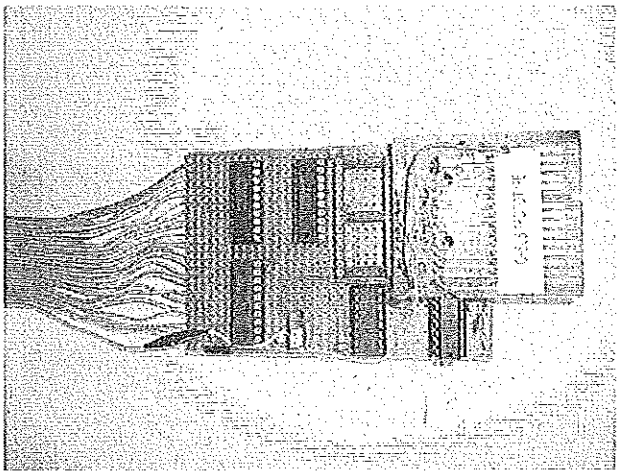
One important feature of the draft constitution is the provision for electronic participation in Quanta meetings including the AGM.

A consultation period of 4 months for constitutional changes is unprecedented in the history of Quanta, but the committee are mindful of the criticism of the 2005 amendments which many say were railroaded through by the former chairman to avoid discussion. They stress the importance

of members making full use of this consultation period as the complexity of a revised constitution means that it will be impossible to vote on it clause by clause at the AGM.

SER-USB - The next Stage?

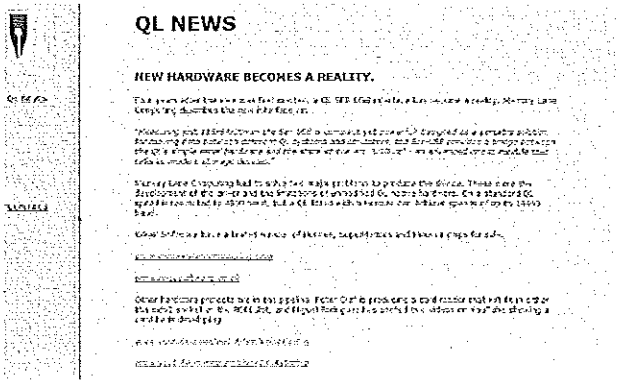
Memory Lane Computing have announced a possible further development of the Ser-USB:
"Whereas the Ser-USB is targeted solely at file transfers, and is limited by the speed of the QL's truly abysmal serial ports, the Ser-USB++ would be a full fixed disk replacement operating at significantly higher speeds.



The Ser-USB++ would use the same core driver (with a different Hardware Abstraction Layer slotted in) and would be targeted solely at Base QLs running QDOS versions up to and including Minerva. It will use the ROM port for its connection and a microcontroller (essentially another IPC outside the QL, but 100s of times more powerful than the QL itself) to provide a standard command interface that will isolate the QL from the underlying protocols of the storage devices that it controls.
Why the ROM port? Because every QL has one, and the expansion slot is often occupied with floppy controllers, Trump Cards, Gold Cards etc. Obviously this will not deliver the same performance as direct connection to the QL's expansion bus, but it worked well enough for the Miracle Hard Disk and the ROMDisQ.
Writing through the ROM port has been done several times before and there's no great technical wizardry involved; just allocate a 256 byte range of addresses for write operations and map the low eight bits of the address bus onto the outbound data bus when that range is accessed.
Instead of creating a ROM port interface dedicated solely to the Ser-USB++, this project would be built upon a new base peripheral called Q-BUS. This card will plug into the ROM port and provide an external bus with 256 individually addressable read/write 8 bit I/O ports. It will also provide operating system extensions to manage the new I/O bus.
The Q-BUS prototype has been constructed using discrete logic, but this will ultimately be replaced with a single PLD to do all the address decoding, latching etc."

Successful new Web Launch

Just Words! has now launched a new website and report about 300 QL visitors in the first month. This exceeded all expectations, and Just Words! has now added a QL news page to the site.
Just Words! has had a website for over 10 years and up till now has always used a free host. Two years ago the original host Lycos Tripod ceased hosting and given the low number of visits to the site Just Words! announced an intention to close it. At the last minute Lycos Tripod transferred the hosting to another firm, Multimania, and Just Words! kept the site open with minimal maintenance. Multimania proved to be a poor host with Norton reporting it to be a major security hazard. Just Words! has now relaunched its site using a paid host and its own domain.
Just Words! announced two reasons for moving to a paid host and relaunching the site. The first was the discovery that the majority of UK QL-ers are neither Quanta members, readers of QL Today or subscribers to the QL-users email group. The other was the many positive developments in the QL community during the last 12 months.
Just Words! has plans to further develop the site.
www.gwicks.net/justwords.htm


QL NEWS

NEW HARDWARE BECOMES A REALITY.

QUANTA Magazine

Quanta has a new editor, Lee Privett. Introducing himself in the current edition of the magazine he writes of returning to the QL scene after an absence of about 12 years. Originally he was an

electronics engineer with published material in electronics and computer magazines. He has now moved into Further Education teaching, specialising in media, multimedia and games development.

QL Today understands that Quanta may have some good news for UK members who opt to receive the magazine electronically. Consideration is being given to giving them a subscription reduction next year equivalent to postage costs. It is hoped that in this way more members can be encouraged to opt for the electronic magazine.

A similar provision already operates for overseas members. Only 1 UK member in 7 has opted for the electronic magazine compared with 1 in 3 overseas members.

Back Online

Just in time to meet QL Today's copy deadline, the Quanta website is back online.

The site has been completely redesigned to give it a more modern and professional appearance. Although provision is made for members to log in with a password, at the time of writing Quanta had made no statement about the future plans for the site and how quickly the members' area will be implemented.

www.quanta.org.uk

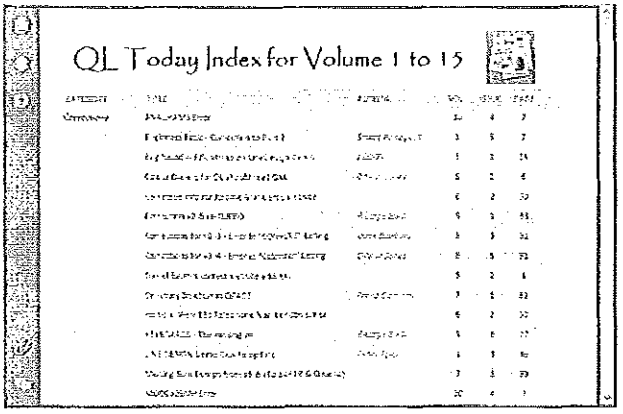
QL TODAY Index

Brian Kemmett has now completed the indexing of QL Today and these can be downloaded from Dilwyn Jones' website. Dilwyn writes:

"Download the indexes either as one index per volume or combined volumes (all as PDF files) from **http://www.dilwyn.me.uk/gen/qltoday/qltoday.html**

Brian has also now supplied a single large PDF file containing the index for all 15 volumes of QL Today magazine. Available from the same page as above. Thanks Brian."

And, of course, a big thank you from QL Today.

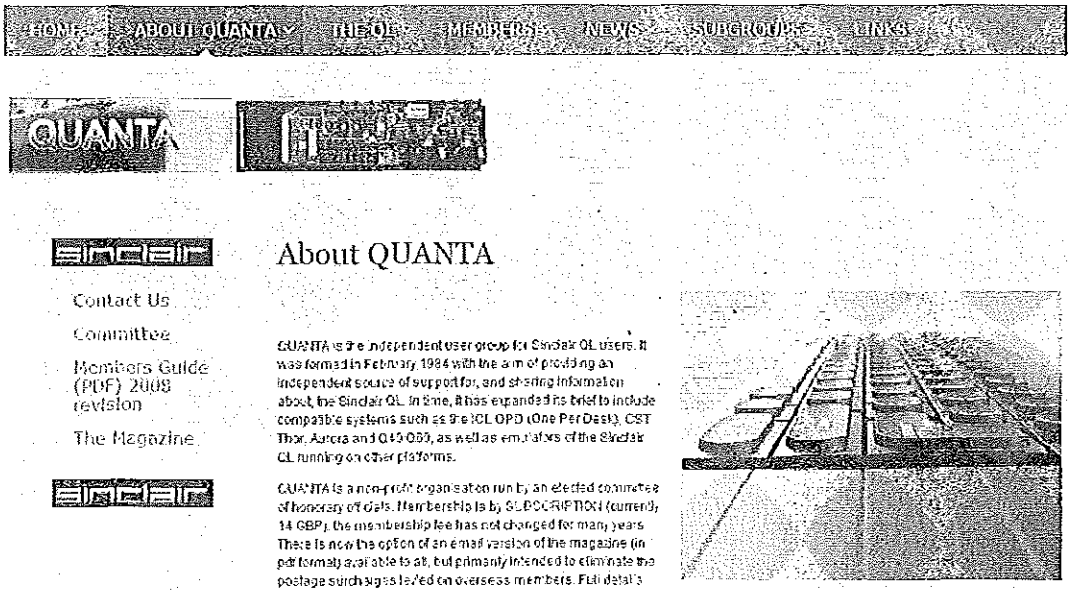

QL Today Index for Volume 1 to 15

HERMES Clarification

Following QL Today's reporting of the SER-USB interface, *Tony Firshman* has issued a clarification of the speeds possible with the Hermes:

"Everywhere the reviews say one can get 19200 with Hermes.

True the baud rate is 19200, but in our tests (years ago now) the maximum input the 8749 could handle was about 14400. The data is 100% OK, but the processor is simply too slow."



SOFTWARE NEWS

ALIEN HIJACK

Rich Mellor writes:

"I have now been able to secure the OK to publish Alien Hijack - the 3D arcade adventure game for the Sinclair QL, originally written by Chisoft and sold under the Maxtronics brand - it can even use their QTALK module.

Alien Hijack places you on-board a ship, where you are the engineer. Having entered the Bermuda Triangle, you find that the ship has been taken over by aliens and the crew vanished.

You have to explore 192 rooms to collect 3 codes, which together form the shape of an anchor, avoiding the various entities inhabiting the rooms. With these codes, you can regain control of the ship.

There is food to replenish your strength, but moving about takes some getting used to."

http://www.sellmyretro.com/offer/details/Sinclair_QL_3D_Arcade_Game%3A_Alien_Hijack-1738

TURBO TOOLKIT

George Gwilt reports that Turbo TK code is now at version 3.39

This version allows COMMAND_LINE to select a daughter SBASIC's channel 0 by giving as a parameter the ID of any channel opened to that SBASIC. The opened channel does not need to be a CON device.

http://gwiltprogs.info/

GD2 XCHANGE

Roger Godley's GD2 versions of Xchange, SILVIA and SOLANO, can now be downloaded from the Just Words! download page:

www.gwicks.net/justwords.htm

Dilwyn Jones has placed a link to the site on the Psions page of his website:

www.dilwyn.me.uk/psions/index.html

The programs can only be run on a resolution of exactly 1024 x 768. Some users have reported problems when scrolling or attempting to modify an Archive program. Unfortunately there is no consistency in the reported problems and QL Today has been unable to reproduce them.

Roberto Orlandi

Davide Santachiara informs us of the death of Roberto Orlandi:

"It is with the deepest sadness and sorrow that we announce the death of our friend Roberto Orlandi. He was 47 years and he suddenly died on 16th of December 2010 while playing tennis. He leaves behind his wife and two children.

We would like to remember his friendliness and the enthusiasm he put to keep the QL passion alive in Italy and abroad. We spent together great years.

Ciao Roberto, you will remain in our hearts forever."

64-Bit Arithmetic

by George Gwilt

S*BASIC on the QL allows two forms of numbers; integers, ranging from -32768 to +32767 and floating point numbers ranging from -(10^616) to +(10^616). If we confine ourselves to integers it would appear that we could have integers with as many as 616 decimal digits. However, the number of significant digits is much less, being limited by the size of the floating point number's mantissa. This has a range of -2,147,483,647 to +2,147,483,648. This is contained in one long word of 32 bits. This is certainly larger than the integers allowed in arithmetic on the QL. Could we somehow perform arithmetic on even larger numbers, say of 64 bits, in S*BASIC? It has been suggested that this could be done by using C68. Here is one method.

C has a type of integer called long long. This consists of 64 bits, just what we are looking for. Given such numbers C68 can add, subtract, multiply and divide them. All we need to do, then, is tell C68, from S*BASIC, what the numbers are and it will produce the answers.

The first step is to find somewhere to store these numbers. This can be done by allocating space from the common heap, by using ALCHP for example. For each number we need 8 bytes. To set space for 10 numbers we need to grab a space of 80 bytes. However, to tell C68 what we want done and where the numbers are we have to communicate with the C program. The obvious way of doing this is to put the information in

a parameter list when the C program is executed. Clearly the address of the area containing the numbers must be part of that parameter list. I suggest that the arithmetical operations and the numbers to be used should be conveyed to the C program, not as further parameters, but inside the area holding the numbers. Thus, I would set an extra four bytes at the start of this area. The first byte would indicate the operation and the remaining three would indicate the numbers to be used. In these three bytes 0 would indicate the first number, 1 the second and so on. If the three numbers are called A, B and C the operation codes would be:

Code	Meaning
0	Add B to C and put the answer in A
1	Subtract B from C and put the answer in A
2	Multiply B by C and put the answer in A
3	Divide B by C and put the answer in A

Putting numbers into the area is not quite straightforward. Although you can POKE a number into one long word and so enter numbers up to 2^32 in size entering larger numbers is not so easy. The method I use is to put the least significant nine digits of the number in one place, C, and the more significant into another, B. The full number can then be entered into A by using operation 127.

127	Multiply B by 10^9 add C and put the answer in A

I have indicated how numbers can be entered into the system. It is obviously necessary to be able to see the results of calculations. We need to be able to translate a number to decimal digits. In a C program this would normally be done using the function printf. Unfortunately printf does not recognise long longs. to rectify this I use a small assembler program which can be loaded into ram and CALLed by S*BASIC. This program has two other useful actions. These are to negate a number and to test it to see whether it is positive, negative or zero. This armoury of operations allows an S*BASIC program to process 64-bit numbers.

The C Program

The listing of the C program is given below. The operations appear as case 0, case 1 etc. In cases 0 to 2 the action is done in the first line. The remaining lines are there to signal errors. For some reason C68 does not check on these errors. For example in a multiplication C68 proudly produces an answer which consists of the last 64 bits only, any more significant bits being ignored. It does this without a blush. So my C program checks the answers. For a multiplication, a division should reproduce the original number. Similarly, for addition and subtraction subtracting and adding should also produce the original. Whenever there is an error detected, the letter E is placed by the C program in the byte originally holding the operation code. This can be checked by S*BASIC.
In the C program "where" is set to hold the address of the area holding the operation codes and numbers. This is filled by using strtoul which is set to enter HEX. Thus the address must be set in the parameter list as HEX.

```
#include‹stdlib.h›
#include‹stdio.h›
#include‹string.h›
#include‹ctype.h›
#include‹qdos.h›
#include‹math.h›

void (*consetup)() = NULL;
char _prog_name[] = "uq";
char *_endmsg = NULL;

int main(int argc,char * argv[])

        {

static    char cop,cd1,cd2,cd3;
static    char* ccop;
static    long long* dub;
static    long int where;
```

```
static    int op,d1,d2,d3,z;
static    long long l1,l2,l3;
          l3 = 1000000000;
          where = strtoul( argv[1],NULL,16);
          ccop=(char *)where;
          cop=*ccop++;
          cd1=*ccop++;
          cd2=*ccop++;
          cd3=*ccop++;
          dub=(long long *)ccop;

          op=cop;  /* op code */
          d1=cd1;  /* where to put */
          d2=cd2;  /* B */
          d3=cd3;  /* C */
          switch (op) {

          case 0:   /* ADD:  B + C  -> A */
                    *(dub+d1) = *(dub+d2)+*(dub+d3);
                    l1 = *(dub+d1);     /* l1 = A */
                    l2 = *(dub+d3);     /* l2 = C */
                    l1 = l1-l2;         /* l1 = A-C sh'd be B */
                    if (l1 != *(dub+d2)) {
                              *(char *)where = 'E';
                              }
                    break;

          case 1:   /* SUBTRACT: B - C -> A */
                    *(dub+d1) = *(dub+d2)-*(dub+d3);
                    l1 = *(dub+d1);     /* l1 = A */
                    l2 = *(dub+d3);     /* l2 = C */
                    l1 = l1+l2;         /* l1 = A+C sh'd be B */
                    if (l1 != *(dub+d2)) {
                              *(char *)where = 'E';
                              }
                    break;

          case 2:   /* MULTIPLY: B*C -> A */
                    *(dub+d1) = *(dub+d2)**(dub+d3);
                    l1 = *(dub+d1);     /* l1 = A */
                    l2 = *(dub+d3);     /* l2 = C */
                    l1 = l1/l2;         /* l1 = A/C sh'd be B */
                    if (l1 != *(dub+d2)) {
                              *(char *)where = 'E';
                              }
                    break;

          case 3:   /* DIVIDE: B/C -> A */
                    if (*(dub+d2))
                     *(dub+d1) = *(dub+d2)/(*(dub+d3));
                    else
                     *(char *)where = 'E';
                    break;

          case 127: /* NEW NUMBER: A = 10^9*B + C */
                    *(dub+d1) = l3**(dub+d2) + *(dub+d3);
                    break;
          default:
                    break;
          }

          return 0;

          }
```

Assembler Program

The listing of the assembler program is given here.

```
; Assume that this is loaded at "asad"

; CALL asad,x - Returns a decimal string for a quad word.
;  x is the address of the quad word.
;  The answer is given in the 20 bytes starting at asad+2.
;  The 1st 2 bytes give the length of the string, len.
;  The string starts at asad+26-len.

; CALL asad+26,x - Negates the quad word at x

; CALL asad+28,x - Tests the quad word at x
;  The byte at asad+2 is -1, 0 or +1 for negative zero or positive.

          BRA.S     START

ANS       DS.L      6
ANS_END   DS.B      0

NEGATE    BRA.S     NEGATE1

TEST      LEA       ANS,A1
          MOVEA.L   D1,A0
          MOVE.L    (A0),D0
          BMI       TEST1
          OR.L      4(A0),D0
          BEQ       TEST2     Zero
          MOVEQ     #1,D0
          BRA       TEST2
TEST1     MOVEQ     #-1,D0
TEST2     MOVE.B    D0,(A1)
          BRA       LB6

NEGATE1   MOVEA.L   D1,A0
          NEG.L     4(A0)
          NEGX.L    (A0)
          BRA       LB6

START     MOVEA.L   D1,A0     -> Quad word . .
          ADDQ.L    #8,A0     . . end
          LEA       STORE+8,A1
          MOVE.L    -(A0),-(A1)
          MOVE.L    -(A0),-(A1)
          SMI       D3        set if negative
          BPL       LB4       positive
          NEG.L     4(A1)
          NEGX.L    (A1)


; STORE now contains the quad word
; The digits are found from the least significnt end and
; placed backwards in ANS

LB4       LEA       ANS_END,A2
          MOVEQ     #10,D2
LB0       MOVEQ     #3,D0
          MOVEQ     #0,D1
LB1       MOVE.W    (A1)+,D1  Next word
          DBNE      D0,LB1    Find 1st non-zero
          BEQ.S     LB3       Finished
```

```
LB2       DIVU      D2,D1      Divide by 10
          MOVE.W    D1,-2(A1)  Set remainder at start of next word
          MOVE.W    (A1)+,D1   Get next word
          DBF       D0,LB2     Count 4 words

; Here D1.TOP contains the next decimal digit

          SWAP      D1
          ADDI.B    #'0',D1
          MOVE.B    D1,-(A2)   Store digit
          LEA       STORE,A1   Reset number
          BRA       LB0
LB3       TST.B     D3         Negative? . .
          BEQ       LB5        . . no
          MOVE.B    #'-',-(A2)
LB5       LEA       ANS_END,A0
          SUB.L     A2,A0
          LEA       ANS,A1
          MOVE.W    A0,(A1)
          BNE       LB6
          MOVE.B    #'0',-(A2)
          MOVE.W    #1,(A1)
LB6       MOVEQ     #0,D0
          RTS                  Return to BASIC

STORE     DS.Q      1
```

S*BASIC Program

The listing below is not a complete program; just a set of procedures and functions that can be used in a complete program or used from the keyboard. The definitions of these are:

Procedures

`Initn%` — This sets the address of an area capable of holding n% 64_bit integers. n% is taken as 6 or more. The value of the area is set in "narea".
The routine uq_bin is set to the address "asad".

`Do_Arithop%,A,B,C` — This performs the operation op%, where op% is one of 0 to 3 and 127.

`Put_Numberno%,n%` — This places no% in the n%+1th 64_bit integer.

`Negaten%` — This negates the n%+1th 64-bit integer.

Functions

`See_Ans$(n1%)` — This returns a string containing the digits of the n1%+1th 64-bit integer.

`Test(n%)` — This tests the n%+1th 64-bit integer. The answer is 0 for OK and -1 otherwise.

Notes

1. The routine 'uq_bin' loaded by Init is the assembled version of the Assembler Program shown above. It appears at the end of Load_Hex as DATA lines.
2. The C program described above appears here as 'ram1_fb'.

Listing of procedures and functions

```
3000 DEFine PROCedure Init(n%)
3005  IF n%<6:n%=6
3010  IF NOT narea:narea=ALCHP(4+n%*8): REMark area for n% 64-bit integers
3015  IF NOT asad:asad=Load_Hex
3020 END DEFine
3030 :
4000 DEFine PROCedure Do_Arith(op%,ans%,a%,b%)
4010  POKE narea,op%
4020  POKE narea+1,ans%
4030  POKE narea+2,a%
4040  POKE narea+3,b%
4050  EW ram1_fb,#1;HEX$(narea,32)
4060 END DEFine
4070 :
4080 DEFine FuNction See_Ans$(n1%)
4090  LOCal x
4100  CALL asad,narea+4+8*n1%
4110  x=PEEK_W(asad+2)
4120  RETurn PEEK$(asad+26-x,x)
4130 END DEFine
4140 :
4150 DEFine PROCedure Put_Number(no%,n%)
4160  POKE_L narea+4+8*n%,0
4170  POKE_L narea+8+8*n%,no%
4180 END DEFine
4190 :
4200 DEFine PROCedure Negate(n%)
4210  CALL asad+26,narea+4+8*n%
4220 END DEFine
4230 :
4240 DEFine FuNction Test(n%)
4245  LOCal k
4250  CALL asad+28,narea+4+8*n%
4255  k=PEEK(asad+2)
4257  IF k=255:k=-1
4258  RETurn k
4260 END DEFine
4270 :
8000 DEFine FuNction Load_Hex
8010  REMark This returns the address of an
8020  REMark Allocationed area containing the HEX
8030  REMark DATA at line 8160
8040  REMark If a mistake occurs -1 is returned
8050  LOCal m,asad,adr,top,x,k,wd%
8060  RESTORE 8160:READ top
8070  IF top<=0:RETurn -1
8080  asad=ALLOCATION(top)
8090  IF asad<0:RETurn -1
8100  k=INT(top/2):adr=asad
8110  m=top-2*k:top=asad+top
8120  FOR x=1 TO k:READ wd%:IF adr+2>top:DEALLOCATE asad:RETurn -1:ELSE :POKE_W adr,wd%:adr=adr+2
8130  IF m:READ wd%:IF adr+1>top:DEALLOCATE asad:RETurn -1:ELSE :POKE adr,wd%
8140  RETurn asad
8150 END DEFine
8160 DATA 172
8161 DATA 24638,0,0,0,0,0,0,0
8162 DATA 0,0,0,0,0,24602,17402,-28
8163 DATA 8257,8208,27402,-32600,4,26374,28673,24578
8164 DATA 28927,4736,24682,8257,17576,4,16528,24672
8165 DATA 8257,20616,17402,102,8992,8992,23491,27142
8166 DATA 17577,4,16529,17914,-62,29706,28675,29184
8167 DATA 12825,22216,-4,26394,-32062,13121,-2,12825
8168 DATA 20936,-10,18497,1537,48,5377,17402,38
8169 DATA 24794,18947,26372,5436,45,16890,-114,-28214
8170 DATA 17402,-144,12936,26120,5436,48,12988,1
8171 DATA 28672,20085,0,0,0,0
```

Lessons from Statistics

by Geoff Wicks

Statistics can be dodgy, but sometimes they are your only source of feedback. I have been snowed under with statistics from the new Just Words! website, and making sense of them has not been easy. However there are lessons to be learnt, particularly for those of us who hold various functions within the QL community such as the editors of magazines, traders and Quanta committee. Some of the statistics have surprised me and in the case of the UK scene give some cause for concern.

Perhaps I can illustrate my difficulties by telling you that in the first four weeks the site had 489 visits and 1224 page impressions, but many of these were not QL-ers. My host, on request, registers the site with the search engines and I had 103 visits and 65 visits from the Google and Yahoo robots respectively. In addition there were visits from Norton, whom I had asked to do a safe site check, and the various bodies that sniff out any new domain. I am fairly certain the four visits a week I get from Beijing are not coming from QL-ers.

I estimate the number of QL visitors to be about 300 and that was far more than I had expected. My old website was attracting only about 9 a week.

The experience of the old site was that the help and advice page attracted the most interest, and that was the page on which I concentrated my efforts in the development of the new site. Eight out of the ten most popular items are help and advice topics. As was my intention the colour tutorial was the most popular item, but, much to my surprise, it held joint first prize with transferring spreadsheets from abacus to a PC.

I first described how to do this in QL Today in 2004 and about a couple of years ago participated in a discussion on the QL-users email group, but clearly this is still a topic on which people want information. Are there other similar topics that need to be placed online or a need for relevant articles in the magazines? Suggestions appreciated.

Another big surprise was the number of hits on transferring LineDesign pages to a PC graphics format. I explained how to do that in QL Today almost ten years ago. At the time traders, including Just Words!, were preparing their adverts in LineDesign and sending them as hard copy by post to Jochen. Quite a few people said transferring LineDesign pages to a PC was impossible, but with a little help from Progs I learnt how to do it. From then on all my adverts were prepared in LineDesign, transferred to gif format and sent to Jochen by email.

Are people still actively using LineDesign or are they just interested in converting old pages? Could it lead to a revival in the use of LineDesign? I know when I was preparing adverts I considered a move to a PC graphics program, but I had already mastered LineDesign and had no desire to embark on a new learning process. Do others think the same way?

Another big surprise is the interest in QTYP and related topics. I thought this would be of minority interest, but I wonder whether people have found the QTYP manual too complicated and welcome alternative help. Several people have also accessed the QTYP dictionaries. In the middle of the last decade I spent much time improving the range and quality of QTYP dictionaries. Before then there had been no separate UK and USA English dictionaries; no definitive German dictionary; and two separate Dutch dictionaries, both with many errors. Dietrich Bruder was not entirely impressed by my German dictionary, but provided me with valuable help in compiling a Swiss German version.

Incidentally if you are interested in other languages and/or dictionaries you can go to Dilwyn Jones' website:

www.dilwyn.me.uk/diction/index.html

Several people have downloaded databases from the maps page. To draw the maps you will have to use the program in an article on the Mercator Map Projection written by Hugh Rooms (QLT v12 i3 p35). You may also find it useful to refer an article I wrote as background to Hugh's work (QLT v12 i4 p9).

In their present form the databases are difficult to use and I hope, in the not too distant future, to be able to convert them into a more QL friendly form that you can use directly in your own programs. Theoretically this should be possible, but until I start the work I do not know how easy.

My host also provides detailed information on browsers and operating systems and both show an increasing movement by QL-ers away from

Microsoft products. Where browsers were known in 2008 47% of visitors were using Internet Explorer. In the first month of the new Just Words! website 41% were using Mozilla, 31% Internet Explorer, 9% Opera and 6% Safari. Where the Operating System was known 81% were using Windows, 12% Linux and 7% MacOS.

Where the origins of the visitors was known some surprising patterns emerged. In the first month the site had more overseas than UK visitors. This was unexpected as Just Words! software was mainly of interest to native English speakers and until now has had a limited impact outside the UK.

The UK statistics give me the greatest concern as these mirror what is happening in Quanta, where there is a lack of interest in the south of England. Visitors to my website from the north outnumber visitors from the south by 3 to 1. The southern visitors come mainly from London, Essex and the South West. Is active QL-ing slowly disappearing in the South East?

Getting the table of contents of an Audio CD from the Internet under QPC - Part 1

by Wolfgang Lenerz

I was recently working with a (non QL) program that would let me add the names of tracks from an audio CD to a database of mine. For that, of course I needed the name of the CD and of the various tracks. There are some linux (and windows) programs that will let you get that information pretty easily from the internet. These programs work as follows: you put a CD in the drive, the program reads the CD, connects to the internet and then presents you with the name of the disk and the names of the tracks on the disk. There is a site called "freedb.org" that contains a large database with information about an incredible number of CDs, and these programs connect to that site.

I thought to myself: why can't I do that on my QL? Well, of course the old black box QL can't connect to the internet, but SMSQ/E running on QPC can. SMSQ/E running on QPC can also access the CD drive of the PC. (So sorry, this article probably is only of interest for those running QPC....). Once you meet these basic requirements, there is nothing to stop you from trying to get a program that also gets this kind of information off the internet. I thus set out to build one. The result can be downloaded from

www.wlenerz.com/QL Stuff/Download

(attention, this is a linux server so the name is case-sensitive). It's a zip file and thus should have preserved the executable file header - if not, there is also the source code: that's in Basic, It should compile just fine (using QLiberator), except that you will need the QPTR and menu extension toolkits which aren't supplied.

A short description of the program :

The program is a pointer driven program. It needs the menu extensions to be present. When the program opens it shows three windows, the upper window, which contains (apart from the usual ESCape item) four menu items: a "question the CD" item, a freedb item, a clipboard item and a "log" item. The lower window contains the "log" : it can show the entire dialogue with freedb or only the result of the query (i.e. the disc name and the names of the tracks). The middle window contains the name of the disc and of the artist.

The "question the CD item" opens and closes the CD tray, checks whether a disk is in there and then gets the "discid" (more of which later). This is mainly useful if you have changed the disc in the drive, as else, the program won't notice that change and still thinks it has the old disc in the drive.

The freedb item does the hard work : read the disk, make the query string and query the freedb site.

The clipboard item copies the content of the log window to the scrap. Beware, if you don't have the "menu extensions", this will crash the program!

The "log" item determines the content of the log window: if "log" is selected, all of the dialogue with the freedb server is shown, if not, only the result of the query is shown.

The data about the disc is displayed in three columns: the name of the artist, the name of the track and the duration of the track. The "sort" loose items at the head of the columns allow you to sort the display according to each column.

So, to have the program work, put a disc in the drive, use the question the CD button and then the freedb button, and then you get the result – provided of course your computer is connected to the internet...

As I will point out during the rest of the text, the program does have a few shortcomings: mainly, it makes some (dangerous) assumptions along the way. If this was a program that was to be sold, I'd add much more error checking into it. I've pointed the various problem areas out in the text. That said, the program works very well for me and I haven't had any crash with it. Perhaps making the program more error-proof can be left as an exercise for the reader... Moreover, if you don't have the menu extensions, the program will crash.

I'm going to comment on some aspects of this program here. Basically, to get the program running, you need three things

1) get some information off the CD itself
2) make this into some information the freedb server can understand and
3) use that information when connecting to the freedb site.

I - Getting the necessary info from the CD

The freedb site needs to be presented with a «discid» and some ancillary information about the CD. This it then uses to search its database. My program contains just two short procedures to read the info from the CD, knowing that QPC already gives me quite a help as it has some interesting inbuilt keywords.

The QPC SMSQ/E manual also gives some interesting information : for audio CDs, one "track" is a title (a song, a movement of a symphony etc.), and a "frame" is one sector of a CD, knowing that the sector length of an audio CD is 2352 bytes. The QPC manual also states: "Redbook – format: a standard format for direct sector addressing. Sectors are addressed through a time index in the form of a longword formatted as
$00MMSSFF
MM is minutes, SS is seconds, FF is the frame. One second has 44100 (Hz)*2(Stereo) *2(16 bit)/2352(Sector length) = 75 frames." Armed with that indispensable knowledge, I set out to create code to read the table of contents of an audio CD.

The first procedure for that is called read_toc and is as follows:

```
2175 :
2180 DEFine PROCedure read_toc
2185 REMark this reads the toc (table of
     contents) into the toc array
2190 LOCal start%,stop%,lp%
2195   IF NOT CD_ISINSERTED
2200      lp%=warning%("No CD in
          Drive!",0,"Error!")
2205      DIM toc(1,4)
2210      RETurn
2215   END IF
2220   start%=CD_FIRSTTRACK : REMark get
       the nbr of the first track
2225   stop%=CD_LASTTRACK   : REMark and
       that of the last track
2230   DIM toc (stop%+1,4)
2235   toc(0,0)=start%:toc(0,1)=stop%
2240   FOR lp%= start% TO stop%
2245     fill_toc lp%
2250   END FOR lp%
2255   fill_part_toc stop%+1,CD_LENGTH
2260 END DEFine read_toc
2265 :
```

The information about the CD will be contained in an array that is imaginatively called toc (for "table of contents"). That is a five dimensional floating point array. Element (0,0) will contain the number of the first track of the CD, element (0,1) the number of the last track. The content of the other elements will be explained later.

The CD_ISINSERTED function in line 2195 checks whether there is a CD in the drive. If not, it shows a warning window. (This is opened in the warning% function which is a standard function of my standard basic library and which I'm not going to comment upon more here as it's not really relevant to the program's functioning). If it can't find a drive it also DIMs the toc array whereby elements (0,0) and (0,1) will be set to 0. Please also note that, before you use the CD for the first time, under QPC you must use the "CD_INIT" command, which in my program is done in the "init" procedure (which is otherwise uninteresting here).

Lines 2220 and 2225 put the number of the first and last track, respectively, into the local start% and stop% variables. They use the CD_FIRSTTRACK and CD_LASTTRACK functions from QPC. As I said earlier, QPC makes accessing an audio CD easy. The toc array is DIMmed to the number of tracks + 2 (element 0 + 1 additional element at the end). Here the program actually presumes that the number of the first track is 1 and that the number of the last track is higher than that of the first track. Whilst

these are reasonable assumptions, in a 'production' program, one should make a check for that (for example, I haven't tested this with CDs that also contains text areas).

In the loop between lines 2240 and 2250, the rest of the elements (except the last one) of the toc array are filled in by calling the "fill_toc" procedure, which is as follows:

```
2265 :
2270 DEFine PROCedure fill_toc (index%)
2275 REMark this makes a table of content entry
2280 LOCal redbook
2285   redbook= CD_TRACKSTART (index%)        : REMark cd track start time
2290   fill_part_toc index%,redbook
2295   IF index%<CD_LASTTRACK
2300     redbook=CD_HSG2RED(CD_TRACKLENGTH(index%)-CD_RED2HSG(CD_TRACKTIME))
2305     toc(index%,4)=CD_MINUTE(redbook)*60+CD_SECOND(redbook)
2310   END IF
2315 END DEFine fill_toc
2320 :
2325 DEFine PROCedure fill_part_toc(index%,redbook)
2330 LOCal a$,minn,sec,frm
2335   a$= HEX$(redbook,32)
2340   minn=HEX(a$(3 TO 4))
2345   sec=HEX(a$(5 TO 6))
2350   frm=HEX(a$(7 TO 8))
2355   toc(index%,0)=minn                              : REMark minutes
2360   toc(index%,1)=sec                               : REMark secs
2365   toc(index%,2)=frm                               : REMark frame nbr
2370   toc(index%,3)=75*((minn*60)+sec)+frm  : REMark frame offset
2375 END DEFine fill_part_toc
2380 :
```

As parameter, the procedure gets an index into the toc array (the entry to be filled in) which corresponds to the track being examined. In line 2285 the start track of the track is gotten from the CD. This is returned as a "redbook" number, as defined above. This number is fed to a second procedure, "fill_part_toc" which fills in one part of the table of contents. The reason the filling of the toc is split into two parts is that for one element only the part of the table of contents is needed and the program would crash if it tried to fill in the entire toc. Line 2335 converts the redbook number into a hexadecimal string. Lines 2340 to 2350 convert the minutes, seconds and frames into numbers and fill in the array elements (index%,0) to index%(2). Element (index%,3) contains the frame offset of the starting location of the track. This is calculated by computing the number of frames from the number of seconds. Finally, the fill_toc procedure fills in element (x,4) which contains the number of seconds for the track.

Once the loop in the procedure read_toc is finished, the last element of the toc array is filled in with the same information as the earlier elements, but concerning the length of the CD (line 2255) and excluding the length of the track in seconds.

As of now we have correctly filled in the toc array and we are ready to make the query string before we contact the freedb server.

II - Making a freebd query string

When you contact the freedb server, you basically ask it whether it has information about a certain CD. To do this, you have to supply the freedb server with a query string containing the "discid" and some further information about the CD: you simply send the freedb server a string that is formatted as follows: "cddb query discid ntrks off1 off2 ...offx nsecs" where:

cddb query is the literal string "cddb query",
the discid is the CD disc ID number that you have to compute,
ntrks: is the total number of tracks on the CD,
off1 off2, ...offx are the frame offset of the starting location of each track,
nsecs is the total playing length of CD in seconds.

To make this query string I use the following function called make_disc_query_command$:

```
300 :
305 DEFine FuNction make_disc_query_command$
310 REMark this makes the ccd command to query the DB for a disc
315 REMark uses global toc variable
320 REMark the format of the command is :
325 REMark -> cddb query discid ntrks off1 off2 ... nsecs
330 REMark    cddb query:
335 REMark          literal string "cddb query"
340 REMark    discid:
345 REMark          CD disc ID number.  Example: f50a3b13
350 REMark    ntrks:
355 REMark          Total number of tracks on CD.
360 REMark    off1, off2, ...:
365 REMark          Frame offset of the starting location of each track.
370 REMark    nsecs:
375 REMark          Total playing length of CD in seconds.
380 LOCal c$,lp%,s%,sp%,t
385   read_toc                                  : REMark read disc table of contents
390   IF toc(0,0)=0 AND toc(0,1)=0:RETurn "" : REMark read errors so do nothing
395   s%=toc(0,0):sp%=toc(0,1)
400   c$="cddb query "&make_ccd$&" "&sp% &" ": REMark first part of query string +
      discid +ntrks
405   FOR lp%=s% TO sp%
410     c$=c$&toc(lp%,3)&" "                  : REMark offsets
415   END FOR lp%
420   t=toc(sp%+1,0)*60+toc(sp%+1,1)+toc(1,0)*60+toc(1,1) : REMark nsecs
425   c$=c$&t
430   RETurn c$
435 END DEFine make_disk_query_command$
440 :
```

Line 400 starts making the string by setting it to "cddb query" followed by the discid and the number of tracks (=i.e. the number of the last track – again a dangerous assumption made by the program). Wikipedia explains that the "discid" is a 32 bit number (4 bytes) such as : $XXYYYYZZ. It also tells us that the first two digits (labelled XX) represent a checksum based on the starting times of each track on the CD, mod 255. The next four digits (YYYY) represent the total time of the CD in seconds from the start of the first track to the end of the last track. The last two digits (ZZ) represent the number of tracks on the CD.

The "discid" is thus made by the following functions:

```
105 :
110 DEFine FuNction ccdb_sum (n%)
115 REMark this makes a ccdb checksum
120 REMark The checksum is computed as follows: for each starting frame (in
seconds!)
125 REMark of a track one sums up the decimal (!!!) digits. e.g., if a frame is at
130 REMark 1234 seconds, the checksum is 1+2+3+4=8.
135 LOCal rett,lp%
140   rett=0
145   REPeat lp%
150     IF n%<=0:EXIT lp%
155     rett=rett + (n% MOD 10)
160     n% = n% DIV 10
165   END REPeat lp%
170   RETurn rett
175 END DEFine ccdb_sum
180 :
185 DEFine FuNction cddb_discid (start%,stop%)
190 REMark this makes the disciD
195 REMark uses global toc variable
```

```
200 LOCal lp%,n,temp%,t
205   n=0
210   FOR lp%= start% TO stop%
215      temp%=toc(lp%,0)*60+toc(lp%,1)
220      n=n+cddb_sum(temp%)
225   END FOR lp%
230   t=toc(stop%+1,0)*60 + toc(stop%+1,1)
235   n=n MOD 255
240   RETurn SHIFTY(n,t,stop%)       : REMark use extension to shift bytes
245 END DEFine cddb_discid
250 :
255 DEFine FuNction make_ccd$
260 REMark uses global toc variable
265 REMark makes the discID ass a string
270 LOCal lp%,start%,stop%,discid,discid$,temp%
275   start%=toc(0,0)
280   stop%=toc(0,1)
285   discid=cddb_discid(start%,stop%)
290   RETurn HEX$(discid,32)
295 END DEFine make_ccd$
300 :
```

The "make_ccd$" function basically only calls the "cddb_discid" function and returns the result of that as a hexadecimal string. The "cddb_discid" function makes the various bytes (also using the ccd_sum function which makes the checksum mentioned above) and uses a small m/c function to make and shift the final number. So now we have the discid (line 400) and we can continue building the query string. In the loop in lines 405 to 415, we add the track offsets for the track to the string and finally the total playing time in seconds (line 880). The query string is now finished. All that remains is to connect to the internet... and this will be explained in part 2.

More Assembler Discussions

by George Gwilt and Norman Dunbar

ND's response to GG's "Letter to QL Today re. Norman Dunbar's Article on Assembler - Part 28"

[GG] In Part 28 Norman Dunbar deals with the Hit routine in an application window.
A critical instruction in a PE program is that which "reads the pointer". In Norman's program this is:

```
wrptr    jsr       wm_rptr(a2)
```

This routine keeps track of the pointer's position, any clicks of the mouse and key presses.
When a key is pressed WM_RPTR determines whether it is the selection key for a loose item. If it is, WM_RPTR calls its action routine. If the key selects an application window, WM_RPTR sets the pointer to the middle of the application window and calls its Hit routine.
If the key press, not being a selection key, is one of SPACE, ESC or ENTER, or if a mouse click has occurred, WM_RPTR looks to see if the pointer is in an application window and, if so, it calls the Hit routine.
If WM_RPTR finds that the pointer has moved, it determines whether it is inside an application window and, if so, calls its Hit routine.

[ND] The above is all interesting and when I read it, I started to think that George had just explained why, in my testing, I saw two hits when I pressed the TAB key, for example. It certainly makes sense to consider a hit for the keypress and another for the pointer move into the application window.

[GG] By the way Norman says that you don't need a Hit routine. But the definition of WM_RPTR in the manual says: "*The window manager requires all application sub-windows to have hit routines. In the case of a standard format menu in an application sub-window this may be just a direct jump to the WM_MHIT routine.*

```
JMP       WM_MHIT"
```

In fact it does seem to be the case that if there is a zero pointer to an application window Hit routine the window manager does not in fact crash but behaves as if the routine were simply:

```
MOVEQ     #0,D4
MOVEQ     #0,D0
RTS
```

[ND] Maybe, somewhere deep in the bowels of WMAN, the fact that the hit routine is set to zero causes a jump to WM_HIT internally, possibly an undocumented feature?

[GG] However, there may be versions of PE which are not so accommodating. It might be advisable, therefore, to have an actual hit routine even though you don't want it do anything. Mind you, if there are menu items in the application window the only way of accessing these via WM_RPTR is to have a valid Hit routine.
Norman asks why when he presses TAB, which is the selection key for his application window, the Hit routine is called twice. The answer is this. WM_RPTR, when it discovers that TAB has been pressed, sees that it is the selection code for the application window. It then does the two things mentioned above. It sets the pointer to the middle of the application window and it calls the Hit routine. As soon as the Hit routine finishes, WM_RPTR finds that the pointer has moved and is inside the application window and so immediately calls the Hit routine again. So the one key press does cause the Hit routine to be called twice. In a real program, of course, the Hit routine would be expected to perform quite different operations on the two calls. D2 would be -1 on the first call and 0 on the second. I would guess that normally the Hit routine would simply jump to WM_MHIT on the second call but would do something completely different on the first.

[ND] See above, I think you have provided the explanation where everyone on the QL Users mailing list didn't seem to know. Thanks.

[GG] Norman says that pressing ESC causes the program to quit instead of the Hit routine being called. The reason for this is that ESC is set as the selection key for the 'X' loose item. From what has been said above, the loose item 'X' is called wherever the pointer is at the time. So the Hit routine is never called by ESC.
However, if ESC is not set as the selection key for a loose item and the pointer is in the application window the Hit routine will indeed be called. Note that if the program dealing with events presented to it by WM_RPTR exiting quits when it discovers a CANCEL, then the program will quit if ESC is pressed when the pointer is not in the application window but not otherwise.

[ND] Yes, I noted in the text that the reason we never trap the ESC key in out hit routine is simply because it is used as an activating key for a loose item, so it gets to execute the loose item hit routine rather than the application window hit routine.

[GG] The second version of Norman's program certainly produces interesting results. Since the programmer of the Hit routine can access the status area (A1 points to this) I thought it might be useful to see what bytes $12 and $13 of the status area contained so I modified Norman's program accordingly. These bytes are, respectively, the key stroke and key press. I found that when I pressed a letter the key stroke contained either the upper or lower case letter just as I had pressed it. The letter in D2 was of course always upper case.

[ND] Useful. There are all sorts of expansions that could be applied to that little utility. I tried to keep it short so as to fit into the magazine. There was talk a while back about not printing long listings, so I was attempting to be frugal on the code front!

[GG] Here are some minor comments:

- In place of the last two instructions of the routine 'ptrpos' you could have:
```
bra.s     pr_int2
```
thus saving one instruction.
- The instruction at 'pr_int' is never used and so could be eliminated.
- The two instructions at the end of 'pr_int2' could become:
```
jmp       (a2)
```

[ND] All perfectly valid of course. I'm pretty certain that I was intending to do something with the code at 'pr_int', but I obviously didn't.

[GG] Finally, when I tried the program I got fed up with the positioning of its window. It had only one position. So I added the following three instructions in 'no_err' before the branch to 'wrpt':
```
btst      #pt__move,wsp_weve(a1)
          ; A move EVENT? . .
beq.s     wrpt    ; . . no
bsr       move    ; Do the move
```
That is all that is needed to cause CTRL/F4 to initiate a move. This enabled me to use QMON on it more effectively.

[ND] I think I mentioned how useful EasyPEasy was as a collection of routines to write Pointer Driven programs. This addition of three lines of code to allow a MOVE to be processed just goes to prove my point!

Glossary of Abbreviations and Terms Part 1 - 1 to 9, A to C

by Dilwyn Jones and Lee Privett

From time to time, those of us who write for QL magazines have to use abbreviations. If we explained every single one in every single article, this would make the articles very long indeed. We know that there are plenty of less experienced or less knowledgeable users out there, as well as those new to or returning to the QL, so here is a reference guide to those commonly used abbreviations and terms we run into time to time.

16-BIT COLOUR	Means that 16 bits of computer memory are used to store the colour value for each pixel on the display. This allows up to 65536 colours to be used. Current systems able to display this much colour include QXL, QPC2, Q40 and Q60
8-BIT COLOUR	Means that 8 bits of computer memory are used to store the colour value for each pixel on the display. This allows up to 256 colours to be used. This mode only exists on the Aurora replacement QL motherboard and QPC2
ACP	Abbreviation for the Archivers Control Panel software, used as a front end for archiving programs like Zip.
ADSL	Asymmetric Digital Subscribers Line, an internet connection sometimes referred to as Broadband
Address	A memory location. Each memory location has a consecutive number, starting from 0 and running up to a very large number corresponding to how much memory a computer has or can handle.
A/D or A to D	Analogue to digital conversion. A method by which analogue data is converted into a digital form. The opposite of course is digital to analogue abbreviated to D/A or D to A
AH, JM, JS, MG	Abbreviated names given to the various versions of the QL ROM issued by Sinclair. The letters actually refer to the version of SuperBASIC built in. For Minerva, the equivalent is 'JSL1' while SMSQ's SBASIC uses 'HBA'
AI (file extension)	Adobe Illustrator files, as used by Line Design. This abbreviation can also stand for Artificial Intelligence
Algorithm	A description of a routine which ends with a result. For example, a routine to sort a list of numbers.
Alpha Test	A first test phase of newly developed software or hardware, after which the first bugs are found and the process passes to the second stage, called a Beta Test stage (q.v.)
Altkey	An alternative definition applied to a key or combination of keys which, when pressed, generate that action. For example, ALTKEY "p","PRINT" on the QL means that if you hold down the ALT key on the keyboard and tap the p key, it would generate the keyword PRINT. A means of assigning whole strings to a single key, in principle. The Toolkit 2 add-on for a QL has a keyword called ALTKEY from where this term comes. When written as two separate words, ALT key refers to the key called ALT on a keyboard.
Ampersand	The & symbol, used to join together two strings in QL BASIC, e.g. LET a$ = b$&c$
ANSI	American National Standards Institute
API	Applications Programming Interface. A set of routines or utilities provided so that programmers can write their own applications to use that code

Application	Usually means the same thing as the term "program", but while a program is usually a single executable file, an "application" can be a traditional single executable program, but may also be a more complex package made of several executables working together.
ARC	A file compression utility
Arguments	Not fights between QL users, but rather a list of values which follow a keyword. For example, the command PRINT a$,b$,c$ has three "arguments".
Arithmetic Stack	Also known as the RI Stack. Similar in principle to other types of stack (q.v.) on the QL. This is a kind of pile of values used as temporary or intermediate stores during calculations. Values are always put onto the end of the stack and also taken off the end of the stack, so they have to be taken off the stack in reverse order to that in which they were put on.
Array	A type of variable which can hold multiple values of the same type. For example, a numeric array holds a list of numbers. A string array holds a list of strings and so on.
Arrow Keys	Also referred to as the Cursor Arrow Keys. Refers to the four keys on the keyboard with arrows on them.
ASCII	American Standard Codes for Information Interchange. Standard used for ensuring all computers represent characters with the same numeric codes, e.g. a capital A always has the code 65. Usually only applies for character codes up to 127
ASIC	Application Specific Integrated Circuit
Assembler	A "low-level" programming language using a human-readable form of the instructions of the micro processor. Think of it as text version of the number codes which a processor executes. The text form is "assembled" (turned into a list of machine code numbers) by a program called an "assembler". The term Assembly Language may also be encountered – this is synonymous to the term Assembler when referring to the human text-readable forms of the instructions of the microprocessor.
Backslash	The "\" symbol on the keyboard. In computing, this is distinct from the Forward Slash key "/".
Backup	The act of making a duplicate copy of a file or disk, made in case something goes wrong and the original cannot be retrieved
BASIC	Beginners All Purpose Symbolic Instruction Code. Gulp! That mouthful basically (pun not intended) describes the main language understood by the QL. The QL version of BASIC is called SuperBASIC, or if you are using SMSQ, it's called SBASIC
BBS	Bulletin Board System. A computer you can dial up with a modem on your computer. Many have programs you can download, masses of information, or even message systems where you can leave messages on the system for someone else who also connects regularly to the system
Beta	This term refers to an application program during its development at a stage when it has more or less full functionality, but is still being tested and may still have several bugs in it and not yet ready for release. When a program has passed the Beta testing stage, it then becomes a Release Candidate and undergoes a lot of thorough testing (we hope) before its full release.
Binary	Base 2 arithmetic. The individual digits of a number can only be 0 or 1. So the number 2 is written as 10, while the number 3 is 11 and so on. In SBASIC, a binary number is represented by preceding it with a "%" character, e.g. %1111 is the same as decimal 15.

BIT	BIT is an acronym for Binary digIT, A single digit of a computer memory, 8 bits make up a byte, 4 bits are sometimes referred to as a 'nybble' or 'nibble' of memory
Bitmap	A graphical representation of an image, stored as a two dimensional map of bits. Each bit (monochrome) or group of bits (colour) represents one pixel or one dot of colour of an image. The QL screen picture is organised as a bitmap, for example.
BIOS	Basic Input Output System, basically a set of commands usually stored in permanent memory instructing the CPU to check connections to various computer hardware
BOOT	A special program or piece of code which defines how a program or computer starts up. To BOOT UP is essentially the same as saying To START UP, except with BOOT there is usually a special significance in that it is usually the name of a special program which starts automatically as the computer starts. On the QL, this is usually a SuperBASIC program
Booting	Starting up a computer (as distinct from kicking it when it doesn't work).
BPS	Bits Per Second, a measure of communication speed on a serial interface, for example when using modems
Broadband	Term used to describe fast connection to the internet
Broadcast	When using the QL network, broadcasting is the term given to when you send a file over the network which can be received by any station which happens to be listening. Broadcasting is done by sending a file to station 0 (e.g. SAVE NET0_0), which means any network station which happens to be listening to station 0 for input (e.g. LOAD NET1_0). QL network station numbers (64 maximum) are normally numbered from 1 upward, with 0 being reserved for this special "broadcasting" use.
Browser	Program used to look at web pages
BSI	British Standards Institution
BTW	By The Way, an abbreviation used by bulletin board and email enthusiasts, one of a number of commonly used abbreviations for such phrases
Buffer	An area of memory used to store input or output temporarily e.g. whilst waiting to be output to a device.
Bug	Fault or error in software
BYTE	A unit of computer memory. Think of it as one of a large number of slots in which the QL can store small numbers. If you know about binary numbers, a byte can store 8 bit numbers, from 0 to binary 11111111, or decimal 0 to 255. Programs are made up of sequences or patterns of these numbers, and larger numbers are made up of a few of these smaller numbers strung together
C68	Name of a C language compiler program for the QL. This is the C compiler most used by QL users.
Cache	Cache is space on a hard disk or other type memory used by the operating system to use again normally in increase speed of operation
CAD	Computer Aided Design but is also used for Computer Aided Drafting and Computer Aided Drawing.
CAM	Computer Aided Manufacture, many automatic milling and drilling machines take commands from a computer based system that uses drawings created in CAD software to make components, printed circuit boards for example.
Case	Whether a letter is a capital letter (upper case) or small letter (lower case)

CD-ROM	Compact Disk Read Only Memory, medium for saving data permanently.
Channel	A channel number refers to the place where the parameters of a command are to be sent to or taken from, e.g. PRINT #0,"Hello" sends the word Hello to channel number 0, while INPUT #1,a$ asks you to enter something from screen channel number 1. The channel number must always be preceded by a hash ("#") character.
Checkerboard	A dotted symbol which looks like a small version of a chess board.
Client	A program which makes use of a server (q.v.). When viewing web pages, for example, the remote computer holding the web pages is a Server and the browser program actually viewing those pages on your computer is a Client.
Compiler	A compiler is a computer program that transforms human readable source code of another computer program into the machine readable code that a CPU can execute. For example, the Turbo compiler on the QL turns a SuperBASIC program into a machine code program which the QL's 68008 processor can execute directly. Another example is writing a program such as a C program in a text editor then running it through a program like C68 to turn the "text" to an executable machine code.
Compression	Trying to make a file smaller, to reduce the space it occupies on your computer system, or to reduce the time the file takes to download from the internet.
CON	Console window. A type of screen window on the QL which you can print information to, and get keyboard information from. If you have opened a CON type window, you can not only use PRINT to write information to the screen, you can also use INPUT to allow the user to type in information in that part of the screen. When the QL is started up, SuperBASIC starts with three CON channels open on the display, which you know as #0, #1 and #2
Concatenate	Act of joining two strings together end to end using the '&' operator, e.g. LET a$="He":LET b$=a$ & "llo"
CONFIG	Program used to configure QL programs which use a block of code called a QJump Standard Configuration Block to hold program default values
CPU	Central Processing Unit, the microprocessor at the heart of a computer
CRC	Cyclic Redundancy Test, used in error checking
CTRL-C	This is a special key-press on the QL keyboard, intended to let you switch between programs which are in memory at the same time. Hold down the CTRL key, and tap the 'c' key. This process of switching between programs is called Task Switching
CTS	Clear To Send, an RS232C signal pin

Small Ads

Small ads again after a very long time, at least one!

Maybe most of you forgot about it, but we still offer free private small ads to our subscribers. If you search for something, or you would like to sell or offer something, just send us a letter or an email with the text.

It should, of course, be QL-related, somehow...

So here it is – the first Small ad this year:

WANTED

Does anyone have the game Fleet Tactical Command by Di-Ren? Will happily pay for the game please contact Peter Scott at

peetvanpeebles@yahoo.co.uk

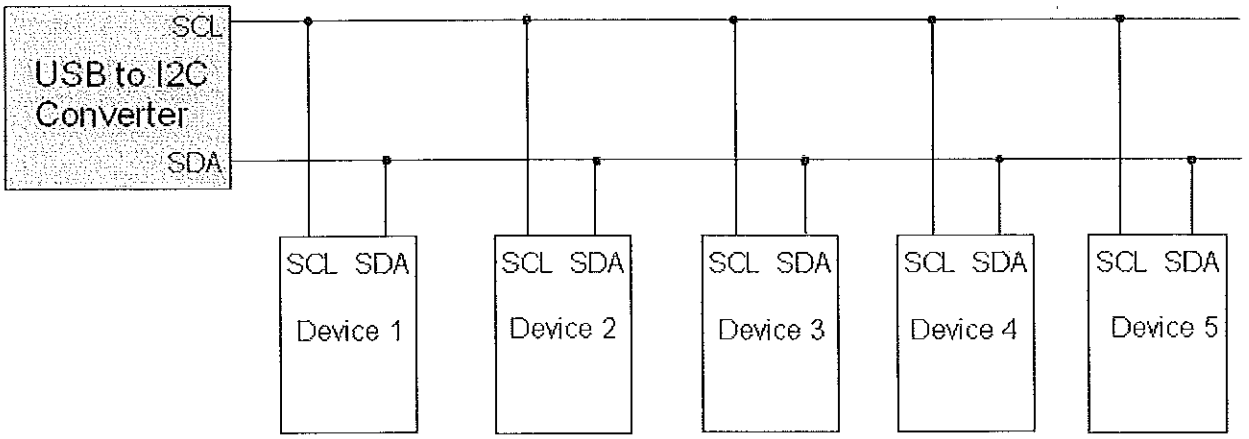
I2C Interface for QL Emulators Part 2

by Ian Burkinshaw

In part one of this series we looked at some of the basics of using the I2C bus and the ByVac BV4221 USB to I2C converter. I also covered the first I2C interface using the PCF8574 parallel device. This time we will look at using the PCF8591 analogue to digital (DA) and digital analogue converter device. The DS1307 RTC (Real Time Clock). The PCF8570 256 x 8 RAM and the DS1803 Dual digital potentiometer we will look at another time.

Last time we really only looked at the I2C in a very simple way, the BV4221 converter driving one device, the parallel (PCF8574) device. However you can have up to 254 devices connected to the I2C bus. So my test/experimental board has one of each of the devices covered in the series of articles. The diagram below shows how this works.

Any number up to 254 devices can be connected to an I2C bus. There being only three connections required. Ground (GND), SDA (Serial Data) and SCL (Serial Clock). An example diagram is shown below to show the connections for multiple devices.

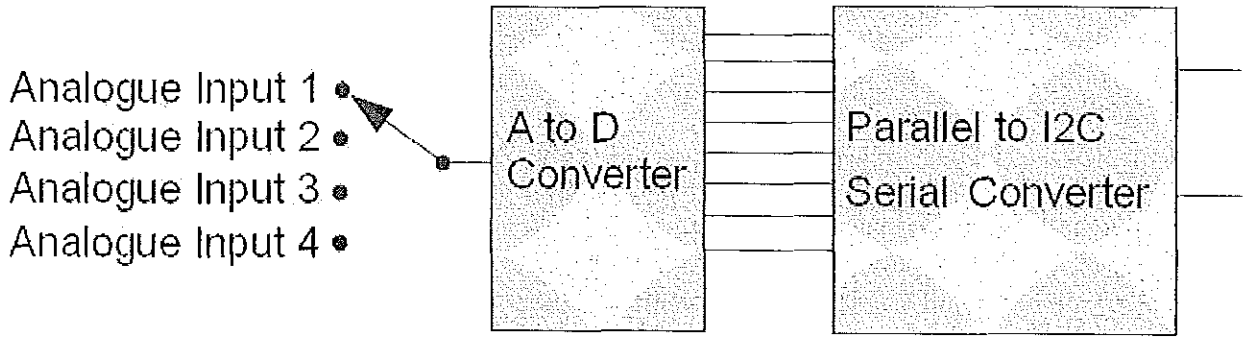


What I have not shown is the fact that 'pull up' resistors or termination resistors if you prefer, are required on the SDA and SCL lines. The reason I have not shown these is they are not required when using the BV4221 converter, since these resistors are already fitted within the converter. The reason for the resistors in the first place is that SCL and SDA are open drain sources. So needs a resistor from the line to VCC power line to complete the circuit. There are reasons that this was done with the I2C protocol which go beyond the scope of these articles. This issue is covered in the I2C book featured below.

So the above diagram shows that you just connect the SCL and SDA lines to all the devices you wish to connect. So you will see that from now on I will not be showing the BV4221 converter in the following circuit diagrams. Just refer back to part one of this serial of articles to see how the BV4221 is connected to any device(s).

Now we will look at the A/D (Analogue to Digital) and D/A (Digital to Analogue) converter device PCF8591.

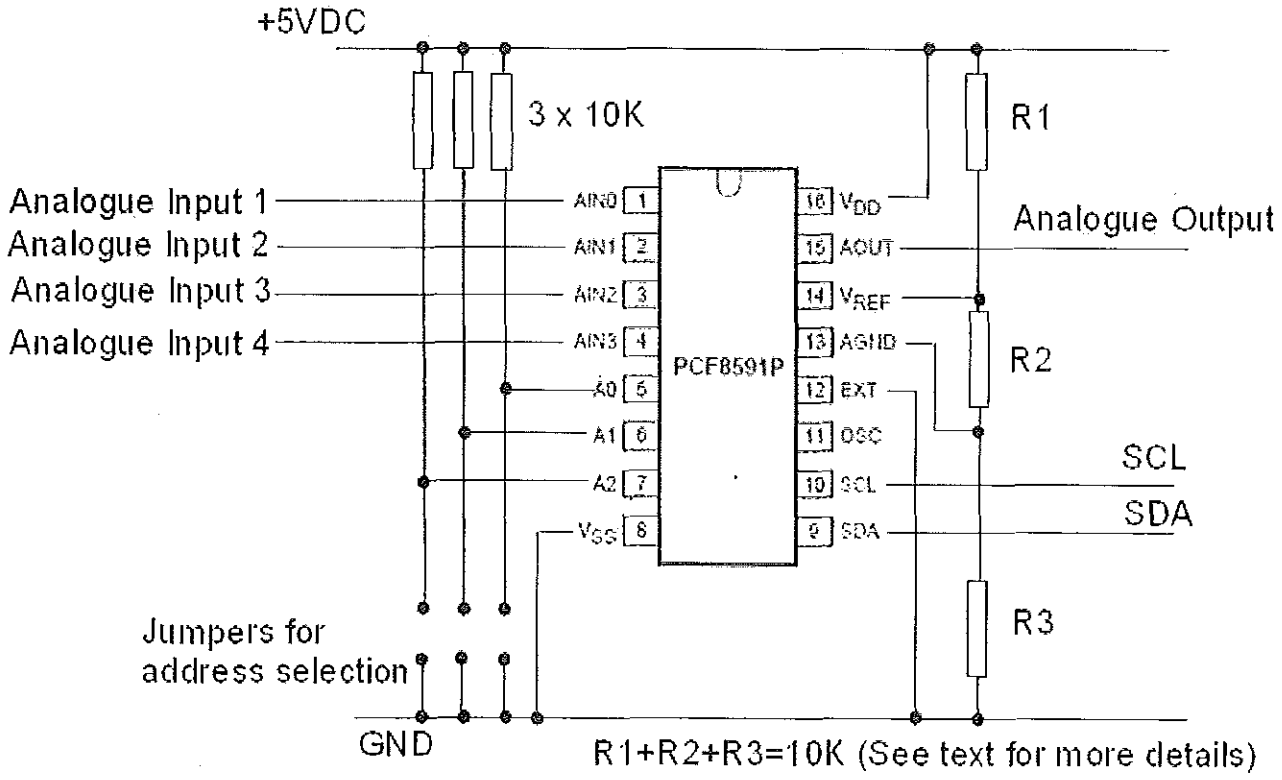
The PCF8591 has a single A(nalogue) to D(igital) converter. But as you see it has 4 analogue inputs. There is a 4 x 1 switch on the input of the device which you can use to select which input you wish to use via the I2C bus protocol for this device. So for example you could have 4 temperature sensors. One connected to each input. Then just select each device one by one, each time you select a sensor take a measurement and then move on to the next sensor and again take a measurement. Working though all the sensors connected and going back the first one and start the process all over again.



Simplified Block Diagram of the A/D side of the PCF8591

The converter is not that fast. The maximum I2C data bus speed is 100kHz, but this is controlled by the master device in our case the QL emulator. So several factors have to be taken into account, for example the baud rate selected from the QL. If you are running a baud rate of 9600 then clearly you cannot drive the I2C bus at 100kHz. Also the number of devices that you are communicating with on the I2C bus in a given application will limit the bandwidth available to the A/D converters. So the sort of applications for this device have to be non time critical. Fine for measuring temperature or voltage several times per second, but not fast enough to record audio signals for example, were you to need to measure at a fairly high and consistent rate. Remember CD's are recorded at 44.1 kHz and 16 bits for example. The resolution of this device is 8 bits so that could be a limiting factor for this sort of application as well.

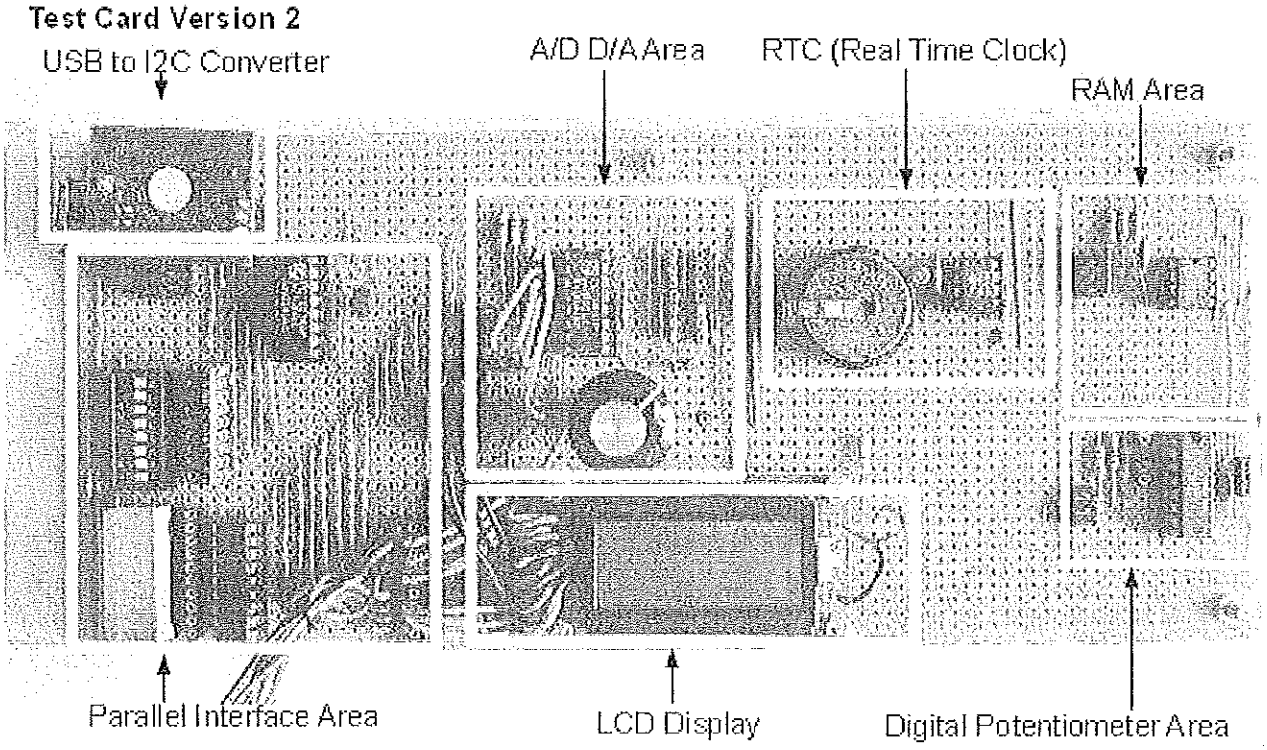
I should point out that Simon Goodwin did produce some software that dealt with audio in a very limited way. Not Hi Fi that is for sure. However this was using the Minerva ROM solution, which will work faster than using the USB to I2C converter. You also have to take into account we are working with an emulator, so we do have to take the PC hardware/operating system into account. It gets very complicated at this level from a timing point of view for this type of application, which is why I have not explored it.



Above you will find the complete circuit of the A/D and D/A converter.

Another detail to take into account is the voltage range of the input is adjustable. Using resistors R1, R2 and R3. These act as a potential divider to set the maximum and minimum voltages. Please note the maximum input voltage can be no higher than 5 volts. Also note the combined resistance of all three resistors should be 10K ohm.

It makes things easy to start with, if you connect pin 14 (Vref) to the 5V rail and connect pin 13 to GND and omit R1, R2 and R3. This sets the converter up to work the range 0V to +5V. Then connect a 10K linear potentiometer, the top connection to the 5V rail, the bottom to GND and the slider to one of the inputs (pins 1 to 4). This will provide you an analogue input to experiment with. In the photo of my test board you will see a control knob. That is the potentiometer I used for testing.



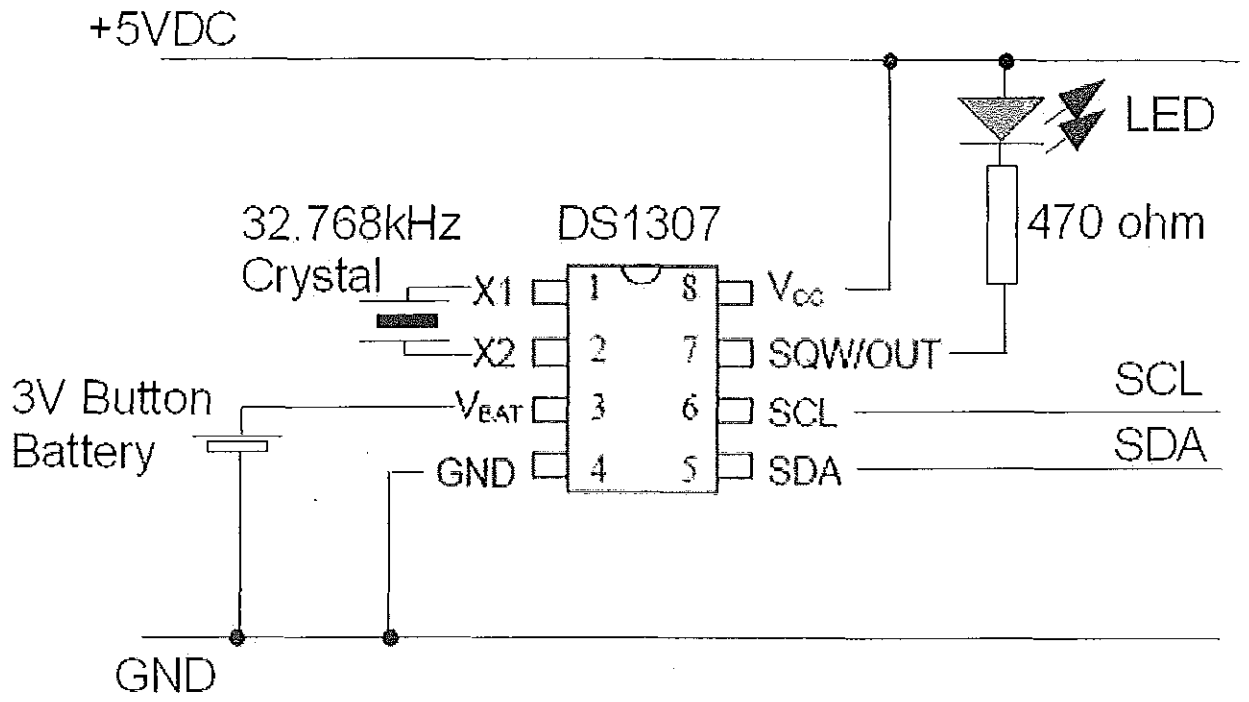
The software for the A/D and D/A device

```
10 CLS
20 BAUD 115200
30 ram=174:REMark PCF8570 address, all address links open.
40 parallel1=126:REMark PCF8574A address, all address links open
50 parallel2=78:REMark PCF8574 address, all links open
60 adda=158:REMark PCF8591 address, all address links open
70 rtc=208:REMark DS1307 real time clock, one fixed address with this device.
80 digpot=94:REMark DS1803 Digital Potentiometer, all link open
90 OPEN#3;ser2ir:REMark i=ignor hardware handshake, r=raw data
100 PRINT#3;CHR$(13);:REMark Carriage Return to set the baud rate in the USB to
I2C converter, required on first pass to initialise USB to I2C converter.
110 print_reply
120 PRINT
130 PRINT#3;"V";CHR$(13);:REMark Command to USB to I2C converter for firmware version.
140 PRINT "Return USB Converter Version Number:-";
150 print_reply:REMark Prints version number reply from USB to I2C converter
160 PRINT#3;"D";CHR$(13);:REMark Sets USB to I2C converter to receive decimal
    numbers, default is hex numbers.
170 PRINT "Decimal Mode Selected"
180 print_reply:REMark returns a device address.
190 DAC
195 AT 10,10:PRINT "          "
200 ADC
```

```
300 AT 10,10:PRINT "End          ":CLOSE#3:STOP
310 :
1000 DEFine PROCedure print_reply
1005 c$=""
1010 REPeat loop
1020 a$=INKEY$(#3)
1030 b$=a$
1040 PRINT b$;
1050 IF a$=">" THEN EXIT loop
1060 END REPeat loop
1070 END DEFine print_reply
1080 :
1090 :
1100 DEFine PROCedure non_print_reply
1110 REPeat loop
1120 a$=INKEY$(#3)
1140 IF a$=">" THEN EXIT loop
1150 END REPeat loop
1160 END DEFine non_print_reply
1170 :
1200 DEFine PROCedure extract_read_data
1205 c$=""
1210 REPeat loop
1220 a$=INKEY$(#3)
1230 b$=a$
1235 c$=c$&b$
1250 IF a$=CHR$(32) THEN slice:EXIT loop
1260 END REPeat loop
1290 END DEFine extract_read_data
1300 :
1310 DEFine PROCedure slice
1320 l=LEN(c$)
1330 IF l<6 THEN GO TO 1350
1340 c$=c$(6 TO l)
1350 END DEFine slice
1360 :
2000 DEFine PROCedure DAC
2010 adcontrol=64:REMark Sets PCF8591 to analogue output enabled, four single
     ended inputs, autoincrement 'Off' and AD channel=0
2020 FOR a=0 TO 255
2030 PRINT#3;"s-";adda;" ";adcontrol;" ";a;" p";CHR$(13);
2040 AT 10,10:print_reply:PRINT a
2050 PAUSE 5
2060 NEXT a
2070 END DEFine DAC
2080 :
3000 DEFine PROCedure ADC
3010 adcontrol=64:REMark Sets PCF8591 to analogue output enabled, four single ended
     inputs, autoincrement 'Off' and AD channel=0
3020 REPeat input_loop
3030 PRINT#3;"s-";adda+1;" g-1 p";CHR$(13);
3040 AT 10,10:extract_read_data:PRINT "      ":AT 10,10:PRINT c$;
3045 REMark FOR b= 1 TO 200000:NEXT a
3050 IF INKEY$=" " THEN EXIT input_loop
3060 END REPeat input_loop
3070 END DEFine ADC
```

The above listing just tests the A/D input and D/A output. To check the output connect a volt meter set to read 5V or above, to pin 15 (AOUT) and GND, when you run the code you will see the output voltage rise. Once the D/A test is finished, then it will test the input, adjust the potentiometer and you should the number displayed on your QPC screen change between 0 and 255.. This is the device working at it's simplest.

Now we will take a brief look at the RTC (Real Time Clock)



As you will see from the above diagram this is very simple. The first thing to note is that there are no address links, this device has only one address, so only one of these devices can be used on a single bus.

You will also note there is a button battery. This keeps the device running when the power it off, so keeps time when not in use. The LED is optional when fully powered up, this blinks once per second.

The software for the RTC

```
10 REMark RTC (Real Time Clock) DS1307 test routines
20 init
30 set_rtc
40 OPEN#3;ser2ir:REMark i=ignore hardware handshake, r=raw data
50 PRINT#3;CHR$(13);:REMark Carriage Return to set the baud rate in the USB to I2C
   converter, required on first pass to initialise USB to I2C converter.
60 print_reply:PRINT "Reply from sending CR."
70 PRINT
80 PRINT#3;"V";CHR$(13);:REMark Command to USB to I2C converter for firmware version.
90 PRINT "Return USB Converter Version Number:-";
100 extract_read_data:PRINT d$:print_reply:REMark Prints version number reply from USB
    to I2C converter
110 PRINT
120 PRINT#3;"D";CHR$(13);:REMark Sets USB to I2C converter to receive decimal numbers,
    default is hex numbers.
130 PRINT "Decimal Mode Selected"
140 print_reply:REMark returns a device address.
150 PRINT
160 PRINT "Writing Clock Set Data"
170 write_rtc
180 PRINT "Writing RAM Area Data"
190 write_rtc_ram
200 PRINT "Both Clock Set Data and RAM Data read from device"
210 read_rtc
220 PRINT "From the first data byte the number should match between what was written to
    the device and what is read from the device."
```

```
230 REPeat loop_time
240 read_rtc_data
250 display_time_date
260 PAUSE 25
270 END REPeat loop_time
280 AT 10,10:PRINT "End          ":CLOSE#3:STOP
290 :
1000 DEFine PROCedure init
1010 CLS
1020 BAUD 115200
1030 ram=174:REMark PCF8570 address, all address links open.
1040 parallel1=126:REMark PCF8574A address, all address links open
1050 parallel2=78:REMark PCF8574 address, all links open
1060 adda=158:REMark PCF8591 address, all address links open
1070 rtc=208:REMark DS1307 real time clock, one fixed address with this device.
1080 digpot=94:REMark DS1803 Digital Potentiometer, all link open
1090 DIM tdata(7)
1100 DIM days$(7,3)
1110 RESTORE
1120 FOR a=1 TO 7
1130 READ d$
1140 days$(a)=d$
1150 NEXT a
1160 DATA "Mon","Tue","Wed","Thu","Fri","Sat","Sun"
1170 END DEFine init
1180 :
1190 DEFine PROCedure print_reply
1200 c$=""
1210 REPeat loop
1220 a$=INKEY$(#3)
1230 b$=a$
1240 c$=c$&b$
1250 PRINT b$;
1260 IF a$=">" THEN EXIT loop
1270 END REPeat loop
1280 END DEFine print_reply
1300 :
1310 DEFine PROCedure non_print_reply
1320 c$=""
1330 REPeat loop
1340 a$=INKEY$(#3)
1350 b$=a$
1360 c$=c$&b$
1370 IF a$=">" THEN EXIT loop
1380 END REPeat loop
1390 END DEFine non_print_reply
1400 :
1410 DEFine PROCedure extract_read_data
1420 d$=""
1430 REPeat data_loop
1440 a$=INKEY$(#3)
1450 b$=a$
1460 d$=d$&b$
1470 IF a$=CHR$(10) THEN EXIT data_loop
1480 END REPeat data_loop
1490 END DEFine extract_read_data
1500 :
1510 DEFine PROCedure set_rtc
1520 PRINT
1530 INPUT;"Year (Last two digits only):-";rtcyear$
1540 INPUT;"Month (number 01 to 12)    :-";rtcmonth$
1550 INPUT;"Date (01 to 31)            :-";rtcdate$
1560 INPUT;"Day (Number ie 1=Mon)      :-";rtcday$
1570 INPUT;"12/24 Hour Clock (Number)  :-";rtchours$
1580 IF rtchours$="12" THEN INPUT;"AM/PM     :-";rtcap$
```

```
1590 INPUT;"Hour (01 to 24)            :-";rtchour$
1600 INPUT;"Minutes (01 to 59)         :-";rtcmins$
1610 INPUT;"Seconds (01 to 59)         :-";rtcsecs$
1620 INPUT;"Pulse output enabled  (Y/N):-";rtcpulseen$
1630 INPUT;"Pulse Frequency (1=1Hz, 2=4.096kHz, 3=8.192kHz, 4=32.768kHz:-";rtcfreq
1640 PRINT
1650 drtcyear=rtcyear$(1):urtcyear=rtcyear$(2)
1660 rtcyear=(drtcyear*16)+urtcyear
1670 drtcmonth=rtcmonth$(1):urtcmonth=rtcmonth$(2)
1680 rtcmonth=(drtcmonth*16)+urtcmonth
1690 drtcdate=rtcdate$(1):urtcdate=rtcdate$(2)
1700 rtcdate=(drtcdate*16)+urtcdate
1710 rtcday=rtcday$
1720 IF rtchours$="12" THEN rtchourt=64
1730 IF rtchours$="24" THEN rtchourt=0
1740 IF rtchours$="12" AND (rtcap$=="AM" OR rtcap$=="PM") THEN rtcap=0
1750 drtchour=rtchour$(1):urtchour=rtchour$(2)
1760 rtchour=((drtchour*16)+rtchourt+rtcap)+urtchour
1770 drtcmins=rtcmins$(1):urtcmins=rtcmins$(2)
1780 rtcmins=(drtcmins*16)+urtcmins
1790 drtcsecs=rtcsecs$(1):urtcsecs=rtcsecs$(2)
1800 rtcsecs=(drtcsecs*16)+urtcsecs
1810 rtcpulseen=0
1820 IF rtcpulseen$=="Y":rtcpulseen=144
1830 rtcfreq=rtcfreq-1
1840 rtcpulse=rtcpulseen+rtcfreq
1850 PRINT "Setting details to be transmitted to device, this is in Hex form"
1860 PRINT "Year:-";rtcyear
1870 PRINT "Month:-";rtcmonth
1880 PRINT "Date:-";rtcdate
1890 PRINT "Day:-";rtcday;" ";days$(rtcday)
1900 PRINT "Hours:-";rtchour
1910 PRINT "Minutes:-";rtcmins
1920 PRINT "Seconds:-";rtcsecs
1930 PRINT "Pulse Output:-";rtcpulse
1940 END DEFine set_rtc
1950 :
2000 DEFine PROCedure write_rtc
2010 PRINT#3;"s-";rtc;" 0 ";rtcsecs;" ";rtcmins;" ";rtchour;" ";rtcday;" ";rtcdate;"
     ";rtcmonth;" ";rtcyear;" ";rtcpulse;" p";CHR$(13);:REMark the first number after
     the s-ram is the starting word address, the remaining numbers are data to be
     loaded incrementing the word address for each data item sent.
2020 non_print_reply
2030 END DEFine write_rtc
2040 :
2050 DEFine PROCedure read_rtc
2060 PRINT#3;"s-";rtc;" 0 p";CHR$(13);:REMark the first number after the s-174 is the
     starting word address, when reading data this set the start word address.
2070 non_print_reply
2080 PRINT#3;"s-";rtc+1;" g-57 p";CHR$(13);:REMark g-9 means it will read 9 data words
     in this example.
2090 extract_read_data:non_print_reply:PRINT d$
2100 END DEFine read_rtc
2110 :
2120 DEFine PROCedure write_rtc_ram
2130 PRINT "Writing Ram Data Only ";
2140 FOR ramd=8 TO 56
2150 dd=RND(0 TO 255):REMark Generating a random number to load into the RAM
2160 PRINT#3;"s-";rtc;" ";ramd;" ";dd;" p";CHR$(13);:REMark the first number after the
     s-ram is the starting word address, the remaining numbers are data to be loaded
     incrementing the word address for each data item sent.
2170 non_print_reply
2180 PRINT dd;" ";
2190 NEXT ramd
2200 PRINT
```

```
2210 END DEFine write_rtc_ram
2220 :
3000 DEFine PROCedure read_rtc_data
3010 FOR a=0 TO 7
3020 PRINT#3;"s-";rtc;" ";a;" p";CHR$(13);:REMark the first number after the s-174 is
     the starting word address, when reading data this set the start word address.
3030 non_print_reply
3040 PRINT#3;"s-";rtc+1;" g-1 p";CHR$(13);:REMark g-9 means it will read 9 data words
     in this example.
3050 extract_read_data:non_print_reply
3060 d=d$
3070 d1=INT(d/16):d2=d-(16*d1)
3080 REMark PRINT dh$;" ";d;" ";d1;" ";d2
3090 tdata(a)=(d1*10)+d2
3100 REMark PRINT tdata(a)
3110 NEXT a
3120 END DEFine read_rtc_data
3130 :
3140 DEFine PROCedure display_tdata
3150 FOR a=0 TO 7
3160 PRINT tdata(a);" ";
3170 NEXT a
3180 PRINT
3190 END DEFine display_tdata
3200 :
3210 DEFine PROCedure display_time_date
3220 AT 42,10:PRINT tdata(2);":";tdata(1);":";tdata(0);"    "
3230 AT 43,10:PRINT days$(tdata(3));" ";tdata(4);"/";tdata(5);"/20";tdata(6)
3240 END DEFine display_time_date
```

This listing sets and reads the RTC.

That is it for this issue, until next time have fun.

For those who would like to delve deeper into the I2C protocol and other devices that can be used, a new book has appeared. It is from the publishers of the Elektor magazine called Mastering the I2C Bus, by Vincent Himpe. It is not a cheap book at £29.50 plus £6.00 for post and packing, prices as at time of writing. But it takes you though from the very basics of the I2C bus to quite advanced projects. It is full of interesting ideas. Also there is a project to provide a USB to I2C converter, this offers an alternative to the ByVac product I have been using in this series. I have not, as yet tried this converter but plan to do so, I will report back in a future article. For more information and to order a copy go to www.elektor.com.

References

http://www.byvac.com/bv3/index.php?route=product/product&product_id=88
(Please note, I used the original V1 of the BV4221, ByVac now supply V2 which also has a SPI interface. The commands are the same, so the programs listed in the article should still work.)
http://www.byvac.com/bv3/index.php?route=product/category&path=44

PCF8570 Ram Data Sheet
http://www.nxp.com/documents/data_sheet/PCF8570.pdf

PCF8574(A) Data Sheet
http://www.nxp.com/documents/data_sheet/PCF8574.pdf
http://focus.ti.com/lit/ds/symlink/pcf8574.pdf

PCF8591 Data Sheet
http://www.nxp.com/documents/data_sheet/PCF8591.pdf

DS1307 RTC (Real Time Clock)
http://datasheets.maxim-ic.com/en/ds/DS1307.pdf

DS1803 Digital Potentiometer Data Sheet
http://datasheets.maxim-ic.com/en/ds/DS1803.pdf

I2C Tutorials
http://www.robot-electronics.co.uk/acatalog/I2C_Tutorial.html
http://www.i2c.byvac.com/ar_foundation.php

TF Services I2C manual
http://www.dilwyn.me.uk/docs/manuals/index.html

Advanced I2C information, but still worth a read to understand I2C protocols
http://www.nxp.com/documents/user_manual/UM10204.pdf

JOCHEN MERZ SOFTWARE

Kaiser-Wilh.-Str. 302 D-47169 Duisburg
http://SMSQ.J-M-S.com SMSQ@J-M-S.com

EPROM MANAGER
You get two main parts: The EPROM Manager & the THING extension. The EPROM Manager makes ready-to-burn EPROM files. You have to give a simple control file and every-thing works automatically. You can put all kinds of files into EPROM: Resident extension (invoked directly or with a new given command), Jobs (executable by command or THING), device etc. Executable files (QRAM, QD, QTYP etc) may be defined to be THINGs and then used from EPROM with HOTKEY II. You can create executable HOTKEYs in ROM now! The current version of the EPROM Manager accepts parameter string. The second part is the THING Extension V3. This gives you the ability to control THINGs from SuperBASIC (and you will learn what THINGs are!!!). Other features: You can test EPROM files in RAM without having to burn them. You can split large files to fit into two or more EPROMs.

QMENU Version 8 and The Menu Extension Version 8
Most Pointer Environment users already know it: the Menu Extension. It is an interface which provides ready-made menus like file-selector boxes, simple-choice-menus or select from a list. QMENU is a guideline how to use it from BASIC, Machine code or maybe other programming languages which allow Machine code interfaces. It explains how to use it with various examples in BASIC and Machine code. You are allowed to use it in your own programs and you may even sell it under license. The Menu Extension also contains the Scrap Extension ("clipboard).
Multi-column menus, file-select with tree and view option, FileInfo II support - just the FileSelect menu on its own is a beatiful extension to your system.
QMENU has not been advertised for quite a while, as the last version 7 manual was not updated in the past few years, while the Menu Extension itself got updated here and there. However, many updates in the Menu Extension and several user inquiries made me think about releasing an updated version of QMENU. The manual has been completely revised and reflects all the minor and major changes and add-ons: from the assembler-side, from the BASIC programming side, and also from the user's side. You get a 42-page printed manual, a floppy disk with updates keys, updated help texts for QD Hyperhelp and updated and new examples.
Please note: The Menu Extension from version 7.65 onwards works only under SMSQ/E V2 (e.g. QPC2 or systems with high-colour screen drivers). If you run the "old" QL Pointer Environment, you should stick to your old Menu Extension. English only (a German version of MENU_rext is also on the disc, but no German documentation). Some menus have got a MOVE facility.
Some of the changes since version 7.04 (the last "officially" documented one) are:
DSEL (Directory Select) allows up to 10 devices
RSTR (Read String) has additional parameters (which force the values entered to be ints, floats, not empty, disables ESC etc) It can also be used to enter hidden passwords.
Timeout feature has been added to RPER (Report Error) and ITSL (Item Select).
New menu SYSS (System select) provides fast selection of items from the Hotkey buffer history, currently running jobs, Things in your system, Executable Things in your system). Just one call and the System Select procedure collects all the information for you and provides it in a list - very easy selection.
Hotkey buffer history now available in the file-select instead of cycling through the "previous" ones.

QPAC I
QPAC I is a collection of very useful pointer-programs: a configurable clock, an alarm which may execute HOTKEYs at a given time, a very comfortable calendar, typer etc.

QPAC II - The pogram everyone needs!
QPAC II is the replacement for QRAM. It does not only add a few more features, it is far superior to it. You can do the file handling, e.g. copy, backup, move, format, create subdirectories etc., control the jobs, things, channels in your machine. Full support of real subdirectories. New features of the curent version: more options configurable, e.g. whether subdirectories should come first in the file list, whether the "Psion, Unlock" menu should be presented on Execute, whether the case of filenames should be kept or converted to lower-case and much more. Adjusts its size much better on hi-res displays.

CueShell
CueShell is a graphically oriented desktop program: program options are represented on the screen and the user has to point at the option only, usually with the mouse, to initiate the desired operation. To copy a file, for instance, the file is marked and then the target is simply pointed to. CueShell is intended to offer easy access to all everyday tasks on the computer.
CueShell is based on the Pointer Environment, a system extension which implements mouse pointer, menu structures and Hotkeys.
Besides being a graphically controlled program, CueShell offers some features not available to QDOS users until now:
Dynamic catalogue window control / object oriented file management, e.g. copying complete directory trees / easy file rename / the form (position, size and sort order) of a catalogue window can be saved and restored for any directory separately / additional file attribute control, write protect and invisible / user-friendly configuration from within the program (saving the current status) / very comfortable view window (scroll files forward and backward, additional options to make _doc files and embedded text in code files easily readable).
See special CueShell offer with QPC!

GAMES FOR SMSQ/E and QDOS (QL)

DIAMONDS/S
This game is a kind of action-puzzle. Keeps you thrilled for a very long time. With High-Score table etc Runs even on 128k QL. New version, now also for other screen resolutions and SMSQ/E. **EUR 11.90**

BRAIN SMASHER!
Not a shoot-'em-up, not an action game. Brain Smasher! is a game which needs your brain. Once started it will keep you busy for some days! You need concentration, tactics and nerves of steel. Very thrilling! It is very difficult to write something about a game ... the price is good, the game is good, why not try it? Needs a memory expansion! **EUR 11.90**

QShang/S
The QL-version of the well-known Shanghai game. For one or two players, many options, very good graphics. Now also improved to run on Minerva, SMSQ/E and high-resolution displays. **EUR 11.90**

The Oracle
An ancient tactic-puzzle, where you do not have to be fast but clever! You have to fill the tiles into the field, but there are various rules how to do it. Every game is different, you'll never be bored! With high-score table, hints for the next move etc.
EUR 11.90

The Lonely Joker Version 2
Twice as many patience games! Yes, a total of six, the new games being Spine, Crapaud and 4 in a Hand. Two of them work with two stacks of cards! Extremely addictive! with High-Score-Table, configurable reverse card sides etc.
EUR 19.90

GAMES FOR QDOS (QL) only

ARCANOID II
Probably the best break-out-type game on the QL. Impressive graphics, 32 levels.
EUR 9.90

FIREBIRDS
Hot-action shoot 'em up. You have to survive many different waves of attack by various opponents. Sometimes up to 32 sprites on the screen. Very fast and not very easy. **EUR 9.90**

ION GOLD & DOPPEL ION
Two German types of coin-machines, very nice graphics and superb simulation.
EUR 14.90

For postage and packing prices please see ad on opposite side!

JOCHEN MERZ SOFTWARE

Kaiser-Wilhelm-Str. 302
47169 Duisburg, Germany

Fax +49 203 502012
EMail: SMSQ@J-M-S.com

SEVERAL PRICE CUTS!

QPC2 Version 3 + SMSQ/E Software QL-Emulator for PC's			**EUR 59,90**
QPC2 Version 3 - Upgrade from QPC2 Version 2			**EUR 19,90**
QPC2 Version 3 - Upgrade from QPC2 Version 1			**EUR 39,90**
QPC Print - printer emulation driver for QPC			**EUR 39,90**
BUNDLE: QPC2 and QPCPrint		**ONLY**	**EUR 79,90**
Agenda Agenda program for WMAN and Prowess	[V1.09]		**EUR 14,90**
Suqcess Database front-end for WMAN	[V2.05]		**EUR 19,90**
QD2003 Pointer-Environment-Editor	[V8.01]		**EUR 29,90**
QD2003 Upgrade from Version 9 and older	[V8.01]		**EUR 14,90**
QMAKE Pointer-driven MAKE for GST/Quanta Assembler	[V4.31]		**EUR 14,90**
BASIC Linker	[V1.21]		**EUR 14,90**
WINED Floppy/Harddisk Sector- & File-Editor	[V1.26]		**EUR 14,90**
FiFi II File-Finder - Extremely useful!	[V4.31]		**EUR 14,90**
FiFi II Upgrade from Fifi Version 3 or older	[V4.31]		**EUR 9,90**
EPROM Manager	[V3.02]		**EUR 14,90**
QSpread2003 Spreadsheet Program	[V4.04]		**EUR 29,90**
QSpread2003 Upgrade from Version 3 and older	[V4.04]		**EUR 14,90**
QPAC I Utility programs	[V1.11]		**EUR 19,90**
QPAC II Files, Jobs & other Things	[V1.45]		**EUR 29,90**
QTYP II Spell checker	[V2.17]		**EUR 19,90**
QPTR Pointer Toolkit	[V0.30]		**EUR 29,90**
DISA Interactive Disassembler	[V3.04]		**EUR 29,90**
CueShell	[V2.14]		**EUR 29,90**
CueShell for QPC	[V2.14]		**EUR 14,90**
SER Mouse software mouse driver for serial mice			**EUR 10,00**
EasyPTR Version 4	[V4]		**EUR 59,90**
EasyPTR Version 4 - Upgrade from earlier versions	[V4]		**EUR 39,90**
QDT - QL Desktop program			**EUR 59,90**
QMENU Version 8 - with new, printed Manual	[V8.02]		**EUR 24,90**
QMENU Version 8 - Update from earlier Versions, also with printed manual			**EUR 17,90**
QMENU Version 8 - New/Update for QL Today subscribers, with prtd manual		**ONLY**	**EUR 14,90**

Please add EUR 4,90 for postage to all destinations - Germany, Europe, Wordlwide!

We accept VISA, MasterCard & Diners Club online and offline!
Details for money transfers:

- Deutschland: Jochen Merz, Account 493 50 431, Postbank Essen, BLZ 360 100 43
- Österreich: Jochen Merz, Account 85055317, PSK Wien, BLZ 60000
- Switzerland: Jochen Merz, Account 60-690080-4, PostFinance, Clearing-Nr. 09000
- The Netherlands: Jochen Merz, Gironummer 3258439, Postbank NL Amsterdam
- and from all other countries in EUR with IBAN and BIC to account Jochen Merz, Deutsche Postbank AG, IBAN: DE21 3601 0043 0611 1004 37 / BIC: PBNKDEFF 360
- UK customers can pay in £ (convert EUR prices above to £ by multiplying with 0.92) to Jochen Merz, Account 83795395, Citibank UK, Sort code 30-00-45 or send cheques in £ - no fee for UK sterling cheques!
- US customers can pay in US$ (convert EUR prices above to US$ by multiplying with 1.48) - no fee for US cheques in US$!

Cheques payable to Jochen Merz only!

Price list on both pages valid until 30th of Oct. 2011

The QL Today DVD 2011

Half a year ago, I did not imagine that this DVD could have become a reality so quickly.

Without the help of Rainer Wolkwitz, Urs König and Dilwyn Jones, it would not exist - **thank you very much for your help!**

This is the first DVD I produced. It should auto-start on Windows systems (unless you turned the auto-start feature off) and should soon show the QL Today logo and several links underneath. If it does not auto-start, have a look at the DVD contents and double-click the index.htm file in the top directory to get it displayed in the browser of your choice. We have chosen Internet Explorer as we know that it is possible to start the QPC demo etc. through it ... that's something we have not tested with other browsers. In case you experience problems, use Internet Explorer (even if you don't like it).

The top level menu

The top two links lead to an overview of the English and German QL Todays respectively.

There is also a lot of bonus material which is either self-explanatory or comes with readme files.

The web links at the bottom require online connection as they lead to pages on the internet... they are there for your convenience.

The QL Today PDFs

If you click on either of the top two links, you will see an overview of all the QL Today issue covers which have been published so far - English and German ones. If you double click a thumbnail, the selected issue should open in a PDF viewer. We do assume here that you have installed a PDF viewer as it is difficult to do something nowadays without having it. There are various suppliers, but in case you do not worry about bulky software, you can download the current version of the Acrobat Reader from **www.adobe.com**

You can navigate backwards and forwards through the menus with the browser's back and forward button.

I hope I have not made any major mistakes in the menus, files or other subjects related to creating a DVD ... as I said, that's my first attempt.
If you find any problems, please write to us. We can't fix it afterwards, but at least we know what to do better next time.
As I write this text, not much time remains before QL Today will be shipped. Copying a DVD for every issue costs a lot of time - it will probably take me a few days in total.
Fingers crossed that all goes well - and if something is not as it should be, we can always have another attempt ... maybe at the end of this volume, we will see. But maybe it's working fine!

Vienna 2012

Gerhard Plaved has informed us that another meeting at Prottes (near Vienna) is planned.

The time to reserve in your agenda is the 7th to 10th of June 2012. It is a "long" weekend, like it was last year: Thursday is a bank-holiday (at least in Germany and Austria).

Together with the suggested date, Gerhard also sent suggestions what could be done on the various days:

Thursday - arrival, meeting at the Donau-Island in Vienna (nice!) and a visit of the Tramway museum in Vienna (nice too, I've seen it already).

Friday - visit of the museum village in Niedersulz.

Saturday - the main day, meeting at Prottes (like last year) ... let's hope the weather will again be so good!

Sunday - last day, no special plans yet ...

We hope to have more details in the next issue - remembering how nice the last meeting was, you should look forward to it!

We assume that more information will also soon be available on his homepage:
www.kuel.org